Technical Report 1125

Summarizing Qualitative Behavior from Measurements of Nonlinear Circuits

Michelle Kwok Lee

MIT Artificial Intelligence Laboratory

This blank page was inserted to preserve pagination.

SUMMARIZING QUALITATIVE BEHAVIOR FROM MEASUREMENTS OF NONLINEAR CIRCUITS

by

Michelle Kwok Lee

May 11, 1989

ABSTRACT

The process of exploring the behavior of nonlinear, dynamical systems can be a time-consuming and tedious process. In this thesis, I have written a program which automates much of the work of an experimental dynamicist. In particular, the program automatically characterizes the behavior of any driven, nonlinear, electrical circuit exhibiting interesting behavior below the 10 Mhz range. In order to accomplish this task, the program can autonomously select interesting input parameters, drive the circuit, measure its response, perform a set of numeric computations on the measured data, interpret the results and decompose the circuit's parameter space into regions of qualitatively distinct behavior. The output is a two-dimensional portrait summarizing the high-level, *qualitative* behavior of the nonlinear circuit for every point in the graph as well as an accompanying textual explanation describing any interesting patterns observed in the diagram. In addition to the graph and the text, the program generates a symbolic description of the circuit's behavior. This intermediate data structure can then be passed onto other programs for further analysis.

This report is a revised version of a thesis submitted on May 11, 1989 to the Department of Electrical Engineering and Computer Science at the Massachusetts Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science.

ACKNOWLEDGEMENTS

Many people have contributed in one way or another to the realization of this thesis. I would like to express my appreciation to Terry Cline and Hal Abelson for their support and guidance on my thesis and for their concern for my intellectual and personal growth. Thanks, also, to Panayotis Skordos for his patience in answering my many questions about signal processing; to John Leo for providing valuable discussions during the early stages of program development; to Mike Eisenberg for reviewing initial drafts of this thesis; and to other the members in my research groups at HP and MIT for their friendship.

Finally, I would like to express my sincerest gratitude to my family, especially my mother and sister. Without their unwavering support, encouragement and love, I would never have achieved what I have. Thank you Mom and Mavis for everything. This report is dedicated to you.

This report describes the research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, supported by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-86-K-0180, and Hewlett Packard Laboratories.

This empty page was substituted for a blank page in the original document.

Table of Contents

List of Figures

This empty page was substituted for a blank page in the original document.

Chapter 1

Introduction

1.1 Goals

This thesis is motivated by the desire to automatically measure the behavior of complex, dynamical systems and to generate high-level, qualitative interpretations of their behavior. In this thesis, I have put together a system capable of automatically measuring the behavior of electrical circuits and performing this high-level interpretation of its behavior. The system takes advantage of information provided by sophisticated measurement instruments and numerical software packages to aid in the identification and analysis of the circuit's behavior.

1.2 Background and Motivation of Thesis

Physicists, mathematicians, and engineers have spent a great deal of time studying the behavior of nonlinear, dynamical systems. They have written numerous papers describing each system's behavior, focusing in particular on the transition route from stability to chaos[DBHL82, Hub83, CYY86, Lin81, KC86, TPJ82, UA81, KKC, MCK85, Sha81]. These papers typically describe interesting behaviors or patterns exhibited by the dynamical system and the parameter values at which they occur. For example, in *Evidence for Universal Chaotic Behavior of a Driven Nonlinear Oscillator*[TPJ82], Testa, Pérez, and

Jefferies discuss the parameter values at which they observe the period-doubling[1] route to chaos in their driven, nonlinear oscillator. Similarly in *The Devil's Staircase*[KKC], Kennedy et. al. trace the progression of the onset of chaos[2] in their transistor circuit; and Chua, in the *Devil's Staircase Route to Chaos in a Nonlinear Circuit*[CYY86], explores the parameter values at which his oscillator circuit exhibits subharmonics of all orders from one to infinity. In fact, the first paragraph describing Chua's experimental results in the Devil's Staircase paper reads,

> As we decrease the frequency f_s from 20 khz to 500 khz, with all other parameters held fixed, we observe subharmonics of all orders from 2 to 33. Each subharmonic is found to persist over a limited range of input frequency f_s. Each [point] in the [parameter-space] graph can be interpreted as a synchronization state between the input frequency and some submultiple of the circuit's *natural* frequency, i.e. the oscillation frequency when the input signal is set to zero. The [area of each region] can therefore be interpreted as the "locking range." [The greater the area; the greater the locking range.]

Aside from providing textual descriptions, investigators have also generated parameter-space graphs which summarize the behavior of a complex, nonlinear system for each point in the two-dimensional grid. For example, Linsay, in his paper *Quasiperiodicity and Chaos in a System with Three Competing Frequencies*[Lin88], presents a series of parameter-space diagrams one of which is shown in figure 1.1, and similarly Hayashi makes use of parameter-space diagrams as shown in figure 1.2[Hay64].

Exploring the behavior of a dynamical system over the range of input parameters clearly provides useful information. Mechanical engineers, for instance, spend much of their time studying the behavior of the mechanical systems they have designed. They

---

[1] A nonlinear system is said to exhibit the *period-doubling route to chaos* if its behavior progresses from subharmonics of order $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow 32 \rightarrow \ldots \rightarrow$ chaos.

[2] Chaos is a commonly observed phenomenon in a wide range of physical systems such as nonlinear electrical circuits, control systems with nonlinear restoring forces, thermal convection in fluids, vibrations in the panels of a supersonic aircraft or rocket, the swings of a pendulum with a vibrating pivot, the rotations of Hyperion, a moon of Jupiter, buckled elastic beams, certain chemical reactions, propagation of light through nonlinear optical devices, the ventricular fibrillation of a heart beat, the firing of neurons in marine mollusk and even the dripping of a leaky kitchen faucet[Moo87].

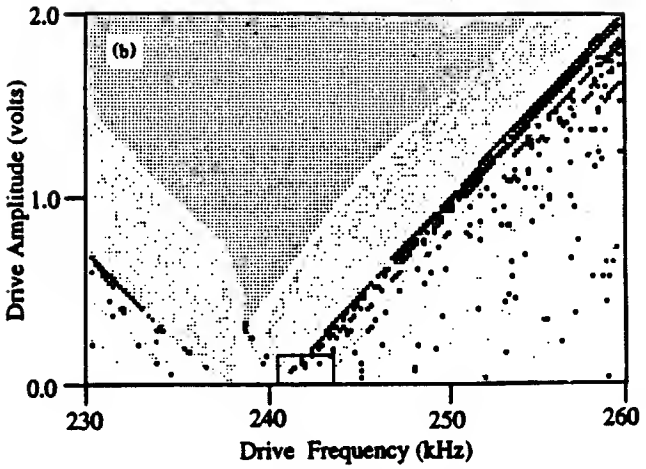

Figure 1.1: Example of a parameter-space graph from Linsay. Points at which the circuit exhibits a periodic response are shown by dots. The white regions indicate quasiperiodic behavior, and the black squares represent chaos.

are typically concerned with locating regions of stability and instability in their devices. Having parameter-space diagrams such as the ones shown in figure 1.1 and figure 1.2 allows the investigator to locate the regions of interest at a glance.

Referring to figure 1.2, we find that at input values of $f = 0.5$ and $A = 10$ (the point labelled 1), the nonlinear system exhibits harmonic behavior whereas point 2, at $f = 2$ and $A = 0.25$, lies on the $\frac{1}{3}$-harmonic and the $\frac{1}{4}$-harmonic border. The diagram also decomposes the system's parameter space into regions of qualitatively distinct behavior. For instance, for values of A and f near the origin, the dynamical system displays brief occurrences of fifth-order and third-order harmonic behavior. For values of A and f far away from the origin, the system exhibits higher order subharmonic behavior such as fifth and sixth.

Exploring the behavior of a complex, nonlinear system and producing a comprehensive parameter-space diagram such as the ones shown in figure 1.1 and figure 1.2 can be a time consuming and tedious process[AS87]. There exist many techniques to help identify a system's behavior, including power-spectrum analysis, tests for periodicity, and the stroboscopic technique. However, the investigator must still repeatedly select interesting values for parameters and initial conditions, run the test, interpret the results, classify the behavior and record it. This must be done for every point in a two-dimensional

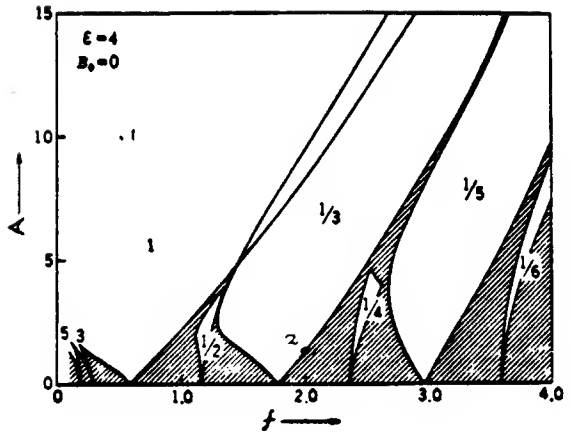

Figure 1.2: Example of a parameter-space graph from Hayashi.

parameter-space. Moreover, in cases where the behavior is not easily identifiable, the investigator must run additional tests to gain different perspectives of the system's behavior. Using the computer to automate this process can greatly aid the investigator in his initial exploration of the system's behavior.

In this thesis I describe a program that can automatically produce a two-dimensional parameter-space diagram characterizing the behavior of a dynamical system. It can set up, run, and interpret measurements for each point in the parameter space. The program can identify four types of behavior: harmonics, subharmonics, quasiperiodics, and chaos.[3] In addition to the graph, the program also produces a high-level, textual explanation to describe any interesting behaviors or patterns in the parameter space of the system.[4] In particular, the textual explanation draws attention to large and small regions of uniform behavior, large regions of chaotic behavior, frequency-locking

[3] Precise definitions of these terms and examples will be given in subsequent chapters.

[4] In the past, scientists like Ueda and Linsay have written programs to automate parameter-space graph generation[UA81, Lin88]. However, their programs did not interpret the high-level behavior in the graph. Instead, they left it up to the user of the program to notice any interesting patterns such as period-doubling cascades. In fact, many of the published papers in the literature of nonlinear dynamics spend a considerable amount of time discussing the investigator's high-level interpretations of the circuit's behavior. Chua's quote on page 10 provides a brief example. The program in this thesis goes beyond the work of Linsay et. al. by *automatically* describing the interesting, high-level, qualitative characteristics of the parameter-space portrait. With such a capability, the program in this thesis could potentially write parts of many of the published papers in the literature of nonlinear dynamics.

ranges,[5] period-doubling cascades and period-adding cascades.[6] It also explains how the behavior of the system varies as the user varies a single input parameter, and it describes the bifurcation curves[7] which border each region of uniform behavior. This is the kind of qualitative, high-level information an investigator seeks when studying the behavior of complex, nonlinear systems experimentally.

Finally, after computing the graph and generating the accompanying textual explanation, the program also returns a symbolic description of the circuit's behavior in the form shown in figure 1.3. This symbolic descriptor, containing information about regions of uniform behavior and boundaries between regions, might then be used by another program to further analyze the behavior of the circuit. For instance, a program might take the symbolic description as input, and it might suggest to the user all the regions in the parameter space in which the device operates in a stable manner. Or it might search the graph for other interesting patterns such as saddle-node, pitchfork, flip or Hopf bifurcations[TS86]. Or it might locate all regions of instability and transition and warn the user against operating near these regions. The program is capable of offering such advice because information about the circuit's behavior is summarized in the symbolic description.

1.3 Sample Run of the Analyzer Program

Figure 1.4 provides a sample parameter-space graph generated by running the program on the forced negative-resistance oscillator circuit[UA81]. Also, accompanying the graph is a text explanation describing the high-level, qualitative properties of the behavior of the circuit. To start the program running, the user needs to type the following input

[5] Frequency-locking ranges are regions in a nonlinear system's parameter-space where the behavior remains constant despite variations to an input parameter. For instance, if we increase the driving frequency of a given circuit from 30000 hertz to 60000 hertz and its response remains fixed at subharmonic order 1, then the behavior of the circuit is said to be "locked" in the range from 30000 hertz to 60000 hertz.

[6] A nonlinear system is said to exhibit a *period-adding cascade* if its behavior goes from subharmonics of order $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow \ldots \rightarrow$ chaos[CYY86]. Compare this definition with the definition of *period doubling* as defined in an earlier footnote on page 1.

[7] A *bifurcation* is a qualitative change in the organization of an equilibrium state as a system parameter is varied.

```
( ;; lumped regions of uniform behavior
  #(((SUBHARMONIC 3)  ;; first region of uniform behavior
     (((VOLTAGE 9) (DRIVE-FREQUENCY 47000))  ;; points in region
      ((VOLTAGE 8.5) (DRIVE-FREQUENCY 47000))))
    ((CHAOTIC)        ;; second region of uniform behavior
     (((VOLTAGE 9) (DRIVE-FREQUENCY 47500))  ;; points in region
      ((VOLTAGE 8.5) (DRIVE-FREQUENCY 47500)))) )

  ;; boundary information
  (((SUBHARMONIC 3 "R1")
    ((CHAOTIC "R2")))
   ((CHAOTIC "R2")
    ((SUBHARMONIC 3 "R1")))) )
```

Figure 1.3: Sample symbolic description returned by the program. The first part of the symbolic descriptor provides information concerning the lumped regions of uniform behavior. The second part presents boundary information.

command:

```
General Format:

(run-point-by-point-2D-analysis circuit-name test-number
                                starting-volt ending-volt volt-units
                                starting-freq ending-freq freq-units
                                volt-delta freq-delta)
Specific Example:

(run-point-by-point-2D-analysis 'neg-resis 4
                                1.5 6.5 v
                                40e3 70e3 hz
                                .25 1e3)
```

where the first argument specifies the name of the circuit under investigation. The second indicates the test number. The third, fourth, and fifth arguments specify the starting voltage, ending voltage and voltage units respectively. Similarly, the sixth, seventh, and eighth arguments pertain to the starting frequency, ending frequency and frequency units respectively. The last two parameter values indicate the incremental step size for voltage and frequency. Thus, in this particular example, the program will analyze the behavior of the forced negative-resistance oscillator over the frequency range of 40000 hertz to

70000 hertz at increments of 1000 hertz and over the voltage range of 1.5 volts to 6.5 volts at increments of 0.25 volts.

Given this information, the program automatically drives the oscillator circuit with a sinusoid of the appropriate amplitude and frequency, sends the appropriate commands to a waveform recorder to record the signal, classifies the response, and produces the parameter-space diagram shown in figure 1.4 along with its accompanying high-level textual explanation shown in figure 1.6. Notice how the textual explanation draws attention to the large and small regions of uniform behavior, the transitions between regions, and any special patterns such as period-doubling cascades. Notice also how the program, like a human, correctly identifies three main regions of uniform behavior—first-order subharmonic, second-order subharmonic, and chaos— in a parameter space that really consists of nine or more regions. This illustrates the program's ability to abstract the regions of uniform behavior, even in a messy graph. In other words, the program is able to take a global enough perspective so as to not let the small, isolated, and sporadic occurrences of third, fourth, fifth, seventh and ninth order subharmonics distract from the dominant first-order subharmonic, second-order subharmonic and chaotic behaviors. As will be discussed in chapter 5 of this thesis, the program accomplishes this abstraction by cleaning up or transforming the "messy" parameter space in figure 1.4 to the new graph shown in figure 1.5.

Thus, the parameter-space graph, along with the textual description, gives the investigator a quick and thorough understanding of the system's behavior, drawing attention to the regions of qualitatively interesting behavior. After running the program once for a first-pass overview, the program can be easily run again, over a much smaller region, to obtain a more detailed picture. Each successive run allows the investigator to zoom in on the region of interest.

1.4 Implications of Thesis Work

The integration of sophisticated measurement instruments with powerful numerical packages and software programs, which interpret the results of physical measurements and numeric computations, suggests new possibilities for test and measurement.

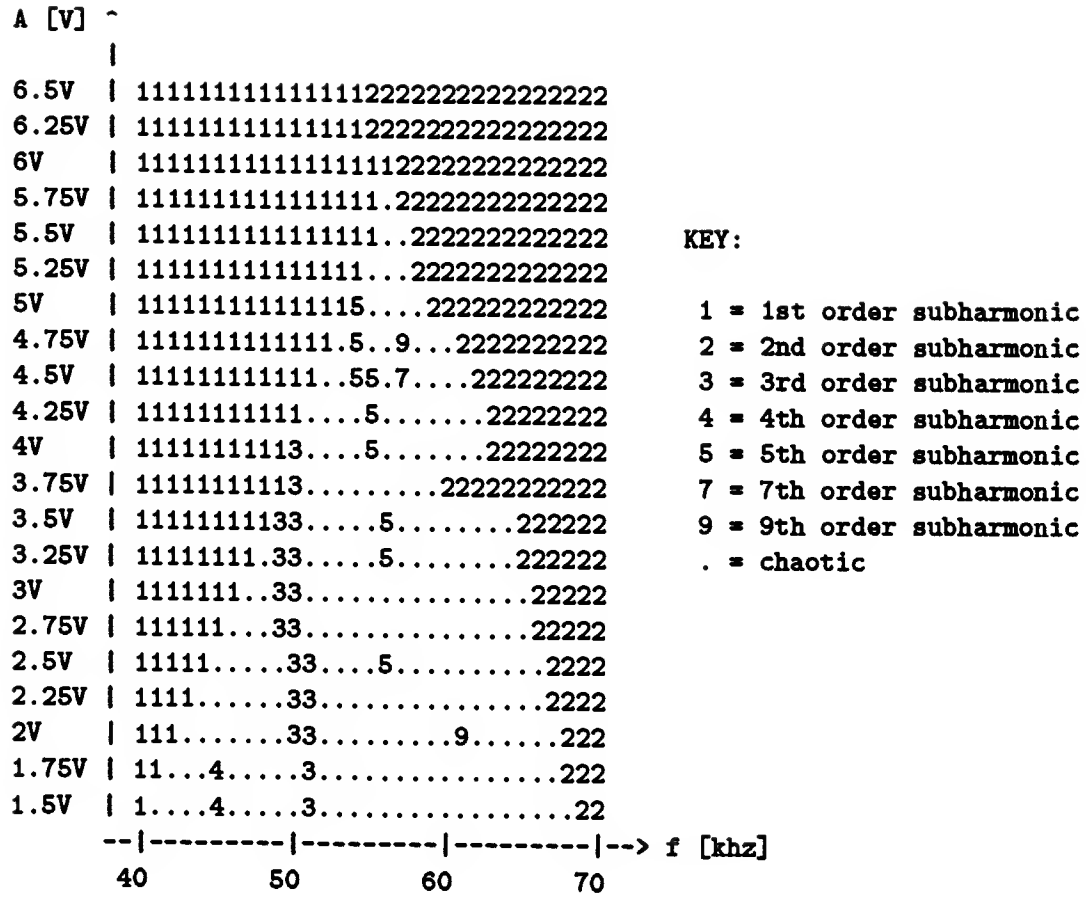

Figure 1.4: Sample parameter-space graph from the forced negative-resistance oscillator circuit. The A along the y-axis indicates the amplitude of the driving sinusoid, while the f along the x-axis specifies the frequency of the driving sinusoid.

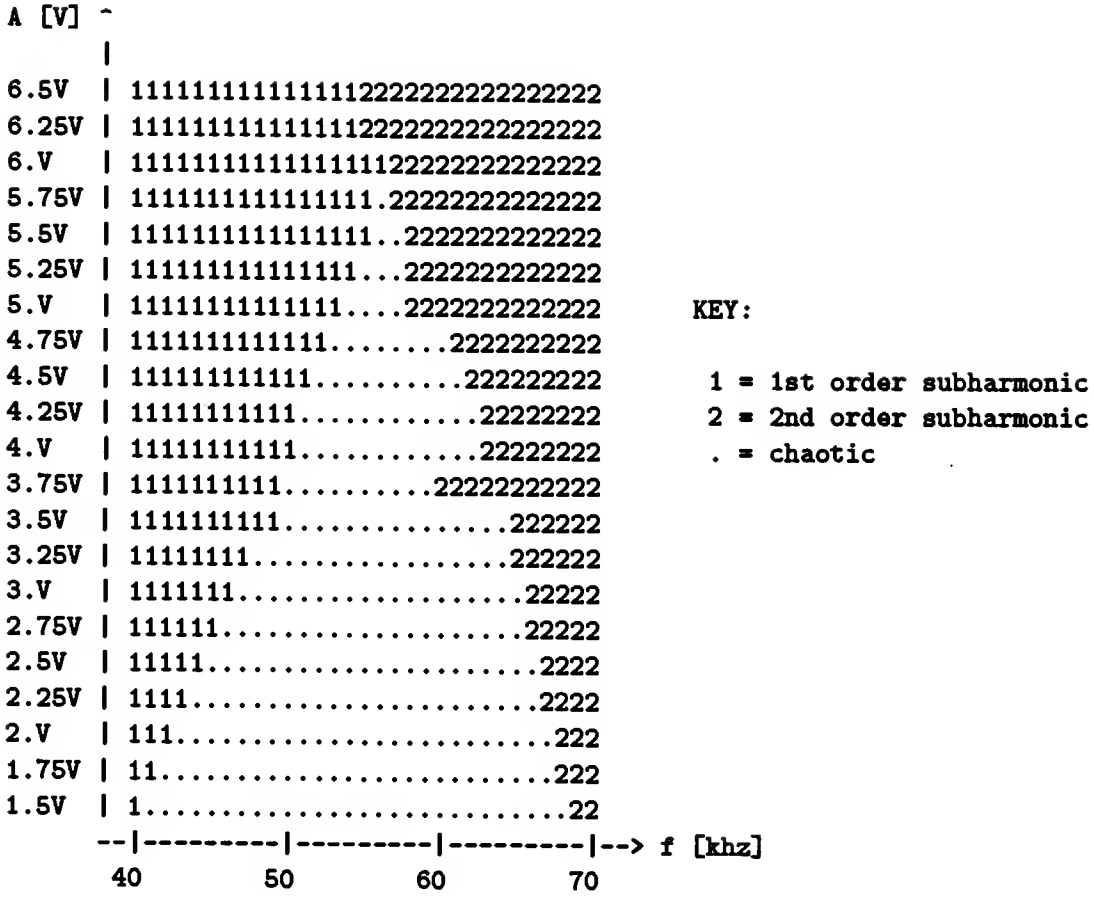

Figure 1.5: Cleaned up parameter space without blemishes.

HIGH-LEVEL INTERPRETATION OF PARAMETER-SPACE DIAGRAM

The parameter-space diagram characterizes the behavior of the forced negative-resistance oscillator as the frequency and amplitude of the driving sinusoid are varied from 40000 to 70000 hertz and from 1.5 to 6.5 volts respectively. For amplitudes between 1.5 and 6.5 volts and frequencies between 40000 to 56000 hertz, the system exhibits predominantly first order subharmonic behavior, while for amplitudes between 1.5 and 5.75 volts and frequencies between 51000 to 68000 hertz, the system exhibits predominantly chaotic behavior, while for amplitudes between 1.5 and 6.5 volts and frequencies between 55000 to 70000 hertz, the system exhibits predominantly second order subharmonic behavior. Aside from these large areas of uniform behavior, the system passes briefly through subharmonics of order 7, 5, 9, and 4 at isolated points throughout the parameter-space graph.

The large region of first order subharmonic behavior is bounded by the bifurcation curve(s) 1-C, and 1-2, where C is a region consisting of primarily chaotic behavior, but littered with sporadic occurrences of other kinds of behavior. Similarly, the large region of chaotic behavior is bounded by the bifurcation curve(s) C-2, and C-1, while the region of second order subharmonic behavior is bounded by the bifurcation curve(s) 2-C, and 2-1.

Finally, based upon the data gathered so far, the forced negative-resistance oscillator shows no clear signs of period-doubling bifurcations as described by Feigenbaum and others.

Figure 1.6: Text generated by the high-level interpreter. In the second paragraph, the notation 1-C indicates a boundary with first-order subharmonic behavior on one side and chaotic behavior on the other. Similarly, the notation 1-2 indicates a boundary with first-order subharmonic behavior on one side and second-order subharmonic behavior on the other side.

Currently instruments play a rather isolated and limited role in the design and exploration stages. Designers or investigators working in most programming environments do not usually use or have access to a wide range of powerful measurement instruments. Even if they do have access to some instruments from their programming environments, the control that they have over an instrument and the results returned by an instrument tend to be fairly low-level. For example, a user might be able to send amplitude and frequency values to set up a function generator, or perhaps a dc-voltage level to a volt meter. Or a dynamic signal analyzer might return several thousand data points as a result of computing the fast Fourier transform (FFT) of a signal. But, interpreting this stream of data points to arrive at some sort of high-level understanding of the behavior still requires a fair amount of human effort.

Aside from the issue of instruments, scientists have typically used computers almost solely for purposes of numeric computations such as convolutions and autocorrelations. The results of such computations are usually in the form of large bodies of numerical data. Although the computer is very good at performing numeric computations, it is not very good at interpreting its qualitative content. It is currently largely up to the investigator to scan through reams of data searching for relevant information.

In this thesis, I have taken advantage of both sophisticated measurement instruments and powerful numerical packages. Yet, on top of these two tools, I have written a program which *interprets* the results of the numeric computations and the physical measurements. As a result, we now have a system with all the power of a whole range of sophisticated measurement instruments, a whole set of numerical packages, and a high-level interpreter, capable of producing meaningful and informative, qualitative, descriptions of complex, nonlinear systems. No longer are scientists limited to low-level, control-the-volt-meter type commands. They now have access to more powerful, more *intelligent* tools, capable of generating such high-level statements as, "The forced negative-resistance oscillator undergoes a period-doubling cascade as the driving frequency is increased from 26 khz to 78 khz while the amplitude is fixed at 5.25 volts."

1.5 Organization of Thesis

Chapter 2, presents a brief overview of the components within this thesis and how they interrelate. Chapters 3 and 4 discuss the details of how the analysis program classifies the behavior of the recorded waveform. In particular, chapter 3 focuses on the use of power-spectrum data for behavior identification, while chapter 4 focuses on the use of time-signal information. In chapter 5, I describe the high-level interpreter which is responsible for recognizing interesting patterns in the parameter-space graph and generating the high-level, textual explanation which accompanies the graph. Finally, chapter 6 concludes with an analysis of the contributions of this thesis and suggestions for future work.

Chapter 2

Overview

This chapter presents the experimental set up of the system as well as a brief description of each component within this system. As illustrated in figure 2.1, the system consists of three main modules: the Observation Equipment, the Measurement Instruments, and the Software Module.

2.1 The Nonlinear Circuit

Before discussing these three components, I would first like to describe the types of circuits the program can handle. Any driven, nonlinear, electrical circuit exhibiting interesting behavior below the 10 Mhz range can be connected to this system for analysis.[1] Moreover, the current system can only have two free parameters: driving amplitude and frequency. The nonlinear circuit shown in figure 2.1 is Ueda's forced negative-resistance oscillator circuit[UA81]. I specifically chose to study this circuit because it provides a rich source of interesting, nonlinear behavior– subharmonics of orders 1 through 19, quasiperiodics, and even chaos. This oscillator is driven by a sinusoid generated by an HP 3325a function generator, and its response is measured over the capacitor and input into the HP 5182a waveform recorder and the oscilloscope. Its nonlinear behavior stems from the nonlinear resistor parallel to the capacitor. Figure 2.2 illustrates the schematic

[1] The 10 Mhz constraint is a limitation set by the HP 5182a Waveform Recorder/Generator. Refer to the specification sheet in the Operating Manual for the HP 5182a.[51884].

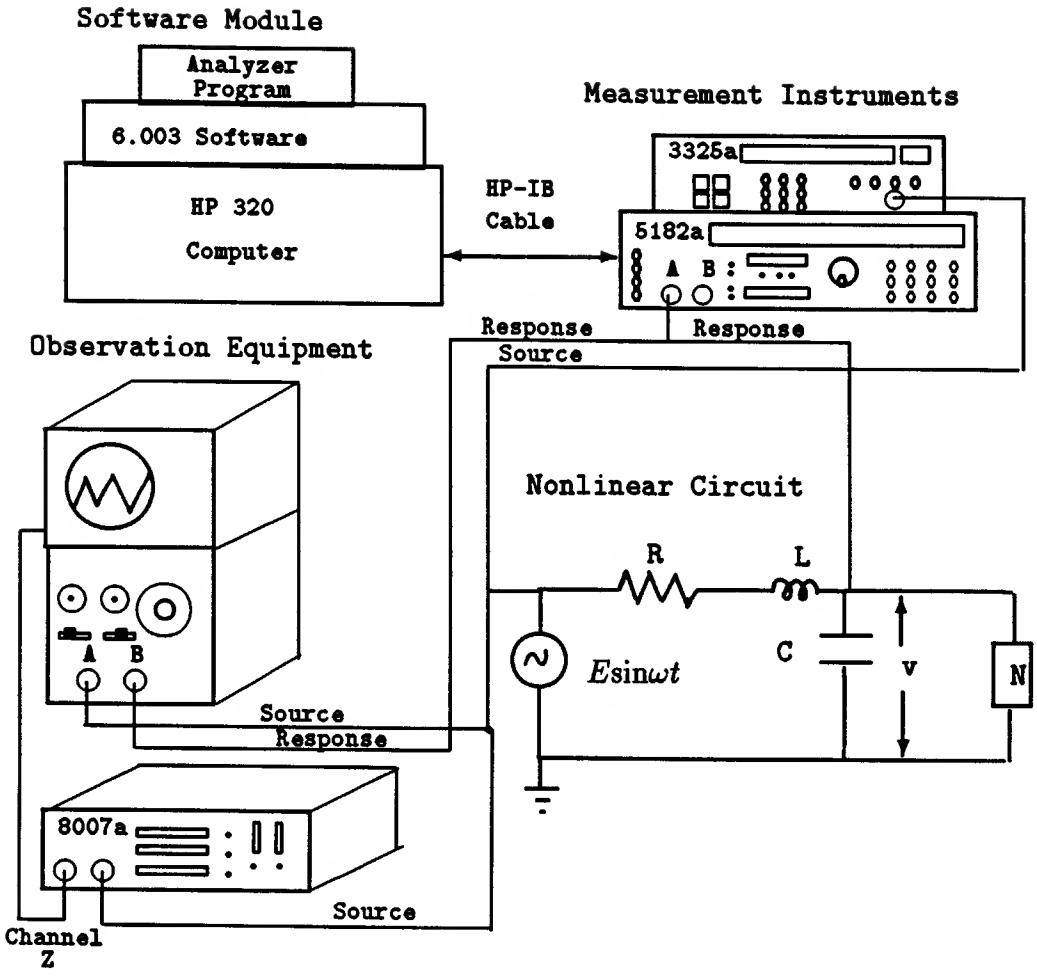

Figure 2.1: Overall layout of the system. The measurement instrument component drives the nonlinear circuit with a sinusoid and records the circuit's response. The software module analyzes and interprets the results of the measurements, and the observation equipment allows the user to visually monitor the behavior of the circuit.

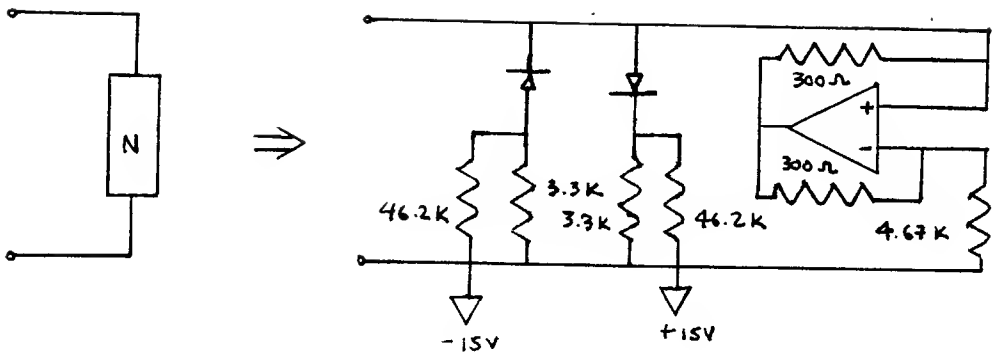

Figure 2.2: Nonlinear resistor in the forced negative-resistance oscillator.

of the nonlinear resistor in the forced negative-resistance oscillator.

In addition to the forced negative-resistance oscillator, we have constructed several other circuits exhibiting interesting nonlinear behavior— Chua's devil's staircase circuit[KKC], Matsumoto's double scroll[MCK85], and a simple, series RLC circuit where the nonlinear element is a varactor diode whose capacitance is a function of voltage. These circuits were connected to the system and used to test the robustness of the analysis program, which was developed primarily using Ueda's forced negative-resistance oscillator circuit.

2.2 The Measurement Instruments

As mentioned earlier, the system uses a set of sophisticated measurement instruments. In particular, an HP 5182a waveform recorder/generator is used to record the response waveform of the circuit under investigation and an HP 3325a synthesizer function generator to produce the sine wave that drives the circuit. For circuits exhibiting interesting behavior under the 100 khz region, the HP 3562a dynamic signal analyzer can also be hooked up to the system for measurement and analysis. However, since most of the interesting behaviors occur at the higher frequencies, the dynamic signal analyzer has been

of limited use to us. All instruments are controlled remotely by an HP 320 computer via HP-IB[2] commands.

2.3 The Observation Instrument

The observation component of the system does nothing to help the analysis program classify the nonlinear circuit's behavior. Its only purpose is to let the user visually monitor in real time the signal and power spectrum of the response. As shown in figure 2.1, the driving sinusoid of the circuit is fed into channel A of the oscilloscope while its response is fed into channel B of the scope. In addition, the output of an HP 8007a pulse generator is connected to the z-input channel at the back of the scope. When the scope is put into $x-y$ mode, we get a Lissajous figure. By counting the number of crests in the Lissajous figure, it is possible to determine the order of the subharmonic[CYY86]. This technique, however, is only practical if the subharmonic order is small enough so the eye can distinguish one crest from another on the oscilloscope's screen. An easier way to determine the subharmonic order by inspection is to strobe the time signal with the pulse generator of the appropriate frequency. If a response is first-order subharmonic, it will display a single highlighted dot in the Lissajous plot. If it is second order, it will have two dots in the Lissajous plot. If the response is chaotic or some very high-order subharmonic response, the Lissajous plot will show a blurring of many, perhaps an infinite number, of dots. Counting the dots is an easy way to determine order subharmonic.

In addition to the oscilloscope, I have input the circuit's response into an HP 3562a dynamic signal analyzer. The 3562a has the ability to compute and display on its screen the power-spectrum response of a signal in real time. Both the oscilloscope and the dynamic signal analyzer provide useful tools to monitor the behavior of the response.

2.4 The Software Module

As shown in figure 2.1, the software module consists of the analyzer program, the 6.003 signal-processing package, and the communication module. The code for all three com-

[2] Hewlett-Packard Interface Bus

ponents resides on the HP 320 computer.

2.4.1 The Analyzer Program

The analyzer program is the heart of the system. It is responsible for setting up the test runs, determining the appropriate parameter values for proper or optimal functioning of the instruments, and sending the appropriate HP-IB commands to the function generator to drive the circuit and to the waveform recorder to record the circuit's response. Once the response has been recorded, the analyzer program transfers the recorded waveform from the waveform recorder to the computer, computes the power spectrum for each response, and classifies the behavior of the circuit into one of four categories: harmonic, subharmonic, quasiperiodic, or chaotic. After having identified the behavior of the circuit for every point in the two-dimensional grid, the analyzer program describes the high-level, qualitative characteristics of the nonlinear system.

Figure 2.3 illustrates the block diagram of the analyzer program. The inputs to the program are data points representing the signal captured from the nonlinear circuit by the waveform recorder, and the outputs include a parameter-space graph, a high-level textual explanation describing the circuit's qualitative characteristics, and a symbolic description to be passed to other programs. The behavior classification module of the system, shown in the top portion of figure 2.3, identifies a system's behavior by examining both the power-spectrum data and the time-signal data. The high-level interpreter component of the system consists of all the boxes below the resolver in figure 2.3. Its purpose is to generate the qualitative text explanation. I will discuss the behavior classification portion of the analyzer program first. This consists of two parts: The power-spectrum analyzer and the time-signal analyzer.

The Power-spectrum analyzer

Once the recorded waveform has been down-loaded onto the computer, the power-spectrum analyzer computes its power spectrum. This power spectrum provides crucial information for the classification of the circuit's behavior. Based upon the location of the peaks in the spectrum, the program can categorize the behavior of the circuit into

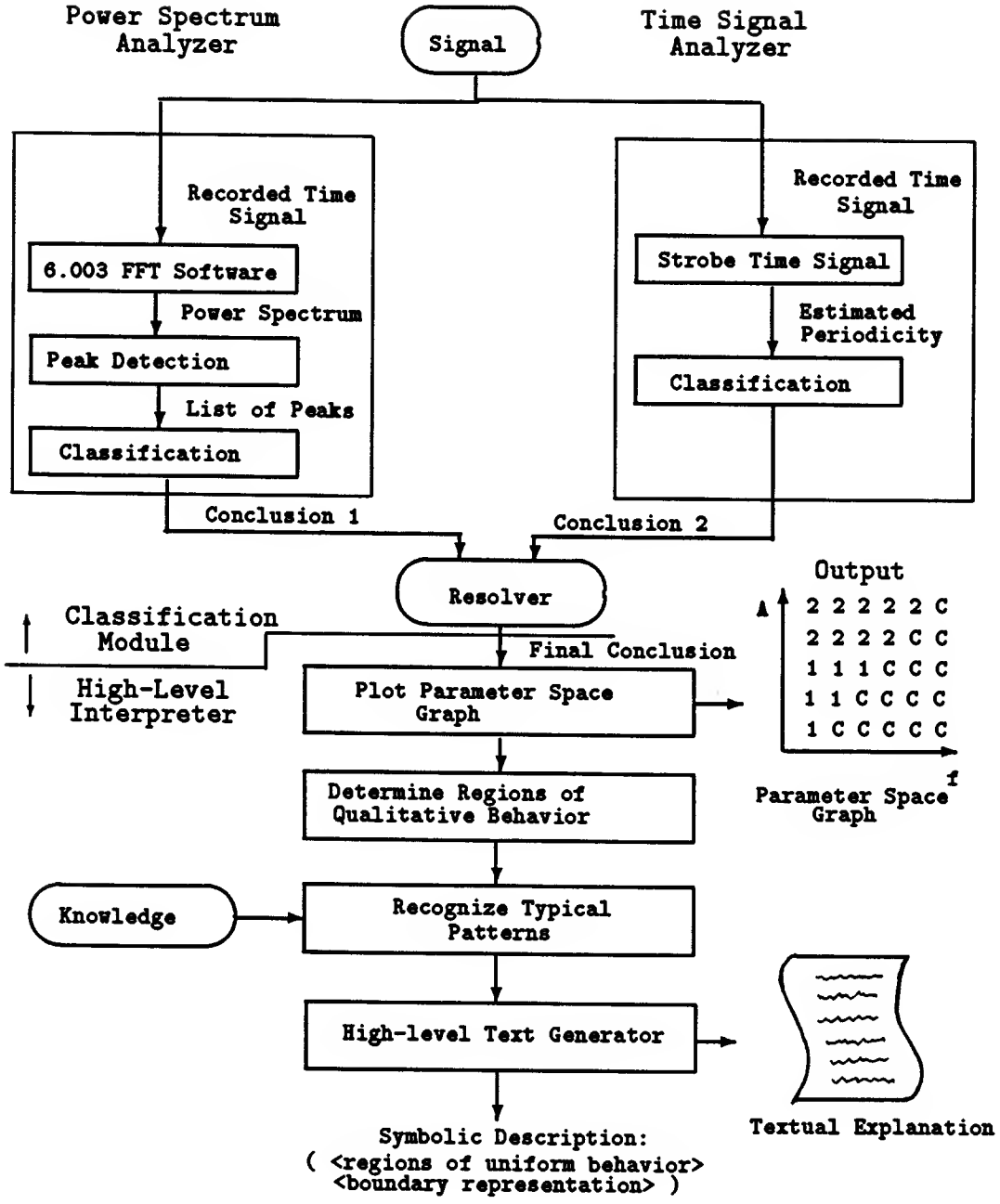

Figure 2.3: Block diagram of the analyzer program.

one of four categories: harmonic, subharmonic, quasiperiodic, or chaotic. Chapter 3 provides an in-depth discussion of the peak detection and the subsequent classification of behavior.

The technique of identifying a system's behavior by searching for peaks in the power spectrum proves effective when the noise level in the spectrum is low. However, as the noise level in the spectrum increases, peak detection becomes more difficult and the accuracy of the power-spectrum analyzer's conclusion degrades. To remedy this problem, the program looks at additional information contained in the time signal to confirm or disconfirm the analysis of the power-spectrum analyzer. By examining the behavior from two perspectives, the program increases the likelihood of at arriving at the correct conclusion.

The Time-signal analyzer

As mentioned in the previous section, the time signal also plays an important role in the identification of the system's behavior. Referring to figure 2.3, we see that the time-signal analyzer strobes the response waveform at multiples of the period of the driving sinusoid. If the response repeats itself with every strobe, it exhibits harmonic behavior. If the response repeats itself every other time, it exhibits second-order subharmonic behavior; if every third time, third-order subharmonic response, etc. As the number of strobes needed for the signal to repeat itself approaches infinity, so too does the order subharmonic. Subharmonic responses whose order approaches infinity are essentially chaotic since by definition a chaotic signal does not ever repeat itself. Chapter 4 discusses in greater depth the details of the time-signal analyzer.

The Resolver

The conclusion produced by the power-spectrum analyzer and the conclusion of the time-signal analyzer should concur. However, if they do not, the resolver module, as shown in figure 2.3, has the responsibility of resolving the conflict. To decide which one of the two conclusions to accept, the resolver looks at *contextual* information. If, for example, the power-spectrum analyzer identified the behavior of a particular point as `CHAOTIC`,

and the time-signal analyzer classified the behavior of the same point as SUBHARMONIC ORDER 2, the resolver will look at the neighboring points to the east and to the south. If one or both are SUBHARMONIC ORDER 2, then the resolver will assume that the power spectrum was extraordinarily noisy and will reject the CHAOTIC analysis in favor of the SUBHARMONIC ORDER 2 analysis. Chapter 4 presents the resolver in greater detail.

The High-level interpreter

Once the program has classified the behavior of the circuit for every combination of input parameters, it produces a two-dimensional parameter-space graph and passes the result to the high-level interpreter. The high-level interpreter scans the parameter-space diagram, first cleaning up any small, isolated occurrences of sporadic behavior in a messy graph, then searching for the boundaries between regions of uniform behavior. Next, the high-level interpreter generates the textual explanations describing any interesting patterns observed in the graph. Knowledge about these patterns is contained in an easily extensible knowledge base. Currently the high-level interpreter can recognize only period-doubling cascades, period-adding cascades and frequency-locking ranges. However, as we add more patterns to the knowledge base, the high-level interpreter will be able to recognize more sophisticated patterns. In Chapter 5, I present some other commonly-observed, well-studied patterns in nonlinear dynamics which we may want to include.

2.4.2 The 6.003 Signal-processing Package

Although the analyzer program makes up a large portion of the software module, the 6.003 signal-processing software also has an important function. This signal-processing environment, developed at M.I.T. for the *Circuits, Signals, and Systems* course, provides a variety of tools for signal analysis, such as fast Fourier transforms, auto-correlations, chirp z-transforms, and convolutions. The primary purpose for including this signal-processing package in the system is to take advantage of its FFT and its autocorrelation procedures.

2.4.3 The Communication Module

The communication module is responsible for translating a high-level Scheme command into the appropriate sequence of HP-IB commands specific to the particular instrument. For example, the analyzer program contains such high-level commands as:

```
(get-time-signal ... )
```

By instrument standards, this is a fairly high-level command. In order to successfully record a signal, the HP 5182a waveform recorder needs a plethora of additional information about the range of the signal, the channel of the incoming signal, whether to AC couple the signal, whether to select automatic sweep mode for triggering, the value for the time base, the memory record length, etc. To make matters worse, each instrument has a different set of HP-IB commands which the user must learn if he wishes to communicate remotely with the instrument. Ideally the user of the system should not have to worry about such details every time he wishes to perform a simple task such as recording a signal. The communication module is smart enough to translate such high-level commands as `get-time-signal` into a sequence of HP-IB commands, with optimally chosen parameter values, to perform the appropriate task.[3] For example, the following code shows the commands generated by the communication module to configure the waveform recorder for `get-time-signal`:

```
;; Select Recording mode as opposed to Generating Mode for the HP
;;  5182a Waveform Recorder/Generator.
  (write-string-to-analyzer "RC" 5182a-device-file)
;; Auto set to certain default values.  See Page 3-52 of 5182a
;;  Operating Manual.
  (write-string-to-analyzer "AU" 5182a-device-file)

;; Select channel A as input.
  (write-string-to-analyzer "CH1" 5182a-device-file)

;; Select Range for channel A.
```

[3] By optimally chosen parameter values I mean the program knows how to, for example, choose the sampling frequency so as to avoid aliasing. More specifically, it chooses the sampling rate so that it is sufficiently below the Nyquist rate.

```
  (write-string-to-analyzer "AR" 5182a-device-file)
  (write-string-to-analyzer (number->string 500e-3)
                            5182a-device-file)

;; Select AC coupling for channel A.
  (write-strings-to-analyzer 5182a-device-file "AC1")

;; Select Sweeping Arming for Trigger to be Automatic Sweep Mode.
  (write-strings-to-analyzer 5182a-device-file "SA2")

;; Set up value for time base.  Note units of time-per-sample
;;  is in seconds.
  (write-string-to-analyzer "MM" 5182a-device-file)
  (write-string-to-analyzer (number->string time-per-sample)
                            5182a-device-file)

;; Set up memory record length.
  (write-string-to-analyzer "LE" 5182a-device-file)
  (write-string-to-analyzer (number->string memory-record-length)
                            5182a-device-file)
  (write-string-to-analyzer "LO1" 5182a-device-file))
```

Chapter 3

The Power-spectrum Analyzer

The purpose of the power-spectrum analyzer module is to classify the system's behavior as chaotic, quasiperiodic, subharmonic or harmonic, based upon information presented in the power spectrum. It achieves this by breaking the task into three main parts: clean up spectrum, detect peaks, and classify behavior.

3.1 Cleaning up the power spectrum

All real power spectra, including those of perfectly harmonic and subharmonic responses, have a certain amount of low-level noise scattered throughout the spectrum. Therefore, before passing the spectrum through the peak detector, the power-spectrum analyzer tries to identify and then to minimize the effects of noise. In figure 3.1(b), I have plotted the power spectrum of a periodic response on a logarithmic scale in order to magnify the presence of the low-level noise in the spectrum.

This noise may be attributed to several factors including:

- *Quantization Error* due to the analog to digital conversion of the signal by the waveform recorder,
- *Arithmetic Error* in the FFT algorithm, and
- *Spectral Leakage* due to a finite-length representation of an infinite-length input signal.

Quantization error results from the representation of continuous signal amplitudes by a fixed number of digital levels within a predefined maximum and minimum analog

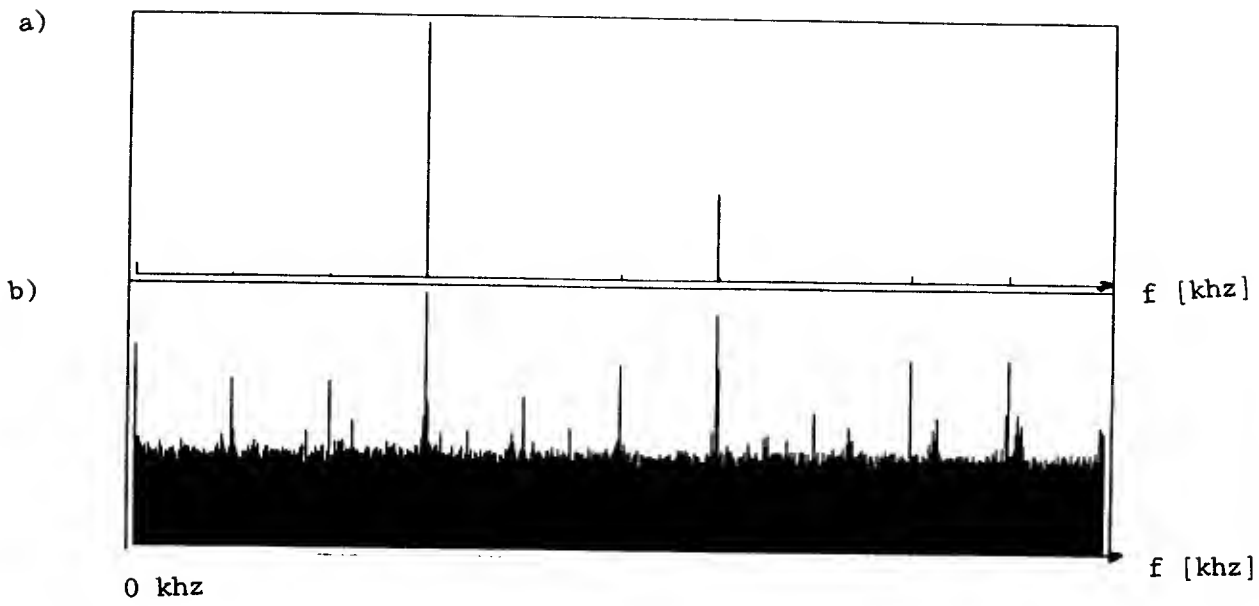

Figure 3.1: Power-spectrum plot to magnify low-level noise: (a) on a linear scale; (b) on a logarithmic scale.

range. Each value of the continuous signal, $x(n)$, must be encoded using B bits to obtain the quantized output, $x^Q(n)$. The quantization error stems from rounding the input value, $x(n)$, to the nearest quantization level, $x^Q(n)$. Figure 3.2 illustrates how the input $x(n) = 0.450$ gets rounded to the nearest quanta, $x^Q(n) = 0.500$.

Thus, the error due to quantization can be represented as

$$e(n) = |x(n) - x^Q(n)| \tag{3.1}$$

It is this $e(n)$ term which contributes to the low-level noise observed in the power spectrum.

In general, the quantization error is a function of B, the number of bits in the A/D converter and the distribution of the input signal over the range of the A/D converter. The greater the number of bits, the smaller each quantum and the less the error.

The peak-detection module minimizes the quantization error by appropriately setting certain parameters on the HP 5182a waveform recorder. The HP5182a with its 10 bits of accuracy,[1] ranges from -511 to $+512$, a span of $2^{10} = 1024 = 512 - (-512)$ points.

[1] For more detailed specification information on the HP5182a waveform recorder, refer to the Operating

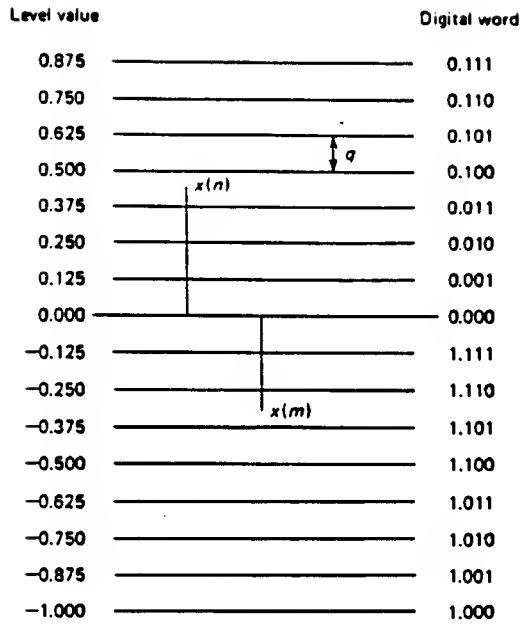

Figure 3.2: Encoding Process Related to A/D Conversion with $B = 4$ and $q = 0.125$ [AN83]

The program automatically scales the amplitude of the input signal so that the incoming waveform makes optimal use of the HP5182a's full range. In other words, regardless of the amplitude of the incoming signal, the program scales the signal so that it occupies nearly the full range of the HP5182a from -512 to +512, thereby minimizing the quantization error.

Spectral leakage results when Fourier transform algorithms try to approximate infinitely long signals with finite-length signals. In practice, one typically has access to only a finite portion of an infinite signal. Computing the fast Fourier transform (FFT) of a finite-length signal has the same effect as taking a finite piece of the signal, replicating it and computing its Fourier transform. If, however, the size of the finite segment does not contain an integral number of cycles, a discontinuity in the signal results, as shown in Figure 3.3[DLH88].

The discontinuity in the signal gives rise to the leakage of sidelobes (or spectral con-

and Programming Manual[51884].

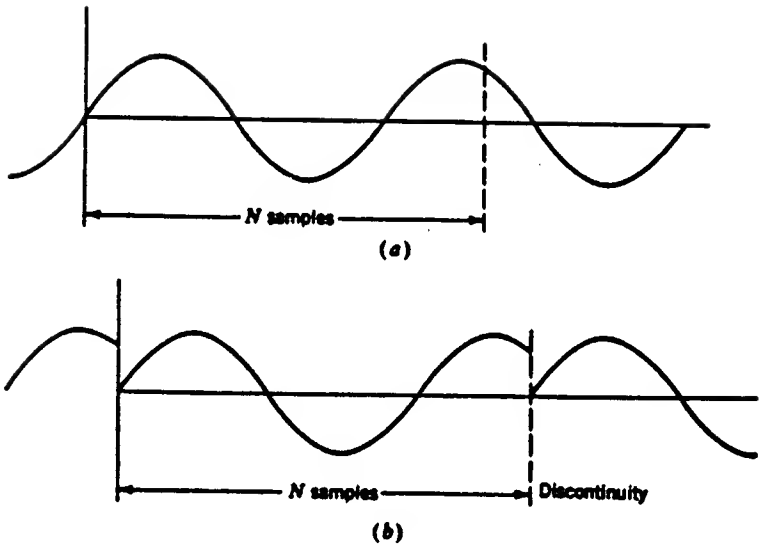

Figure 3.3: Discontinuity resulting from periodic extension of signal by DFT process: (a) continuous signal; (b) signal showing discontinuity due to DFT.

tributions) across the entire frequency set. (See Figure 3.4.) These high sidelobe levels in the spectrum can result in false peak detections. A common way of reducing sidelobe levels is by multiplying the signal by a weighting function[AN83, DLH88, OS75, Har78]. Weighting functions reduce the contribution of the samples near the endpoints, and therefore reduce the discontinuity and its effect on the frequency response. Extensive evaluations comparing the advantages and disadvantages of the various windowing functions ranging from Kaiser-Bessel windows to Gaussian windows have been performed[Har78]. I have chosen to multiply the signal by a Hanning window of order one before taking the FFT. Figure 3.5 illustrates the the power spectra of a windowed input signal. Compare the differences between figure 3.4 and figure 3.5, and notice the reduction in height of the sidelobes due to spectral leakage.

3.2 Peak-detection Component

As mentioned in chapter 1, the key to classifying the behavior of a reponse using power-spectrum data lies in identifying the peaks and recognizing the relationships among their locations. Without an accurate peak detector, the power-spectrum analyzer module has

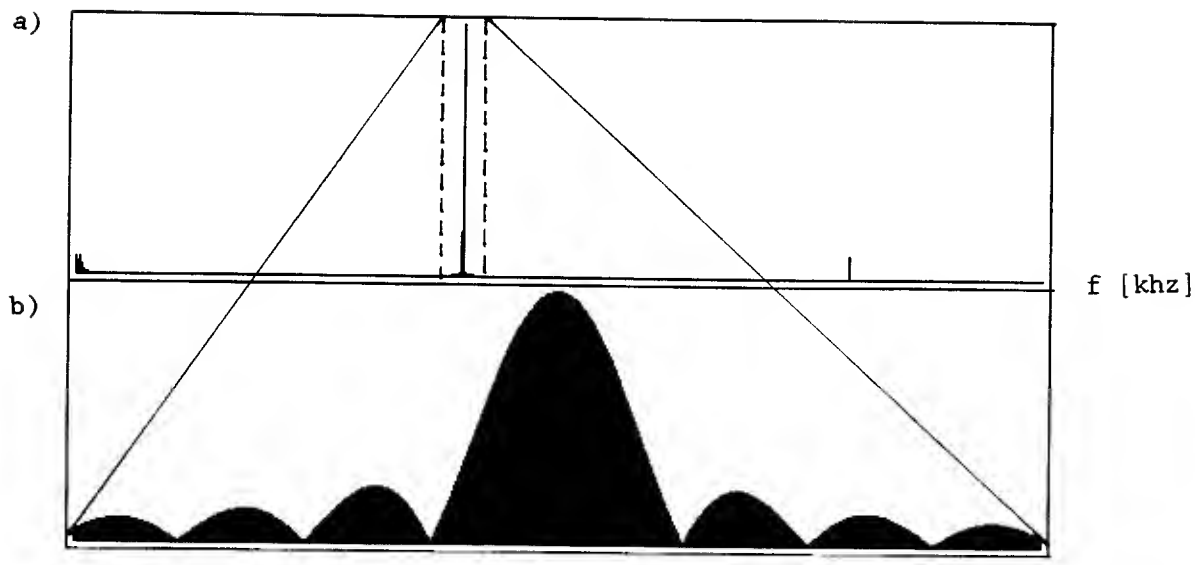

Figure 3.4: Power spectrum of a non-windowed signal. (a) The top graph illustrates the full spectrum on a linear scale. (b) The bottom graph shows a magnified version of (a) around the peak to emphasize the effect of the sidelobes on the low-level noise throughout the spectrum.

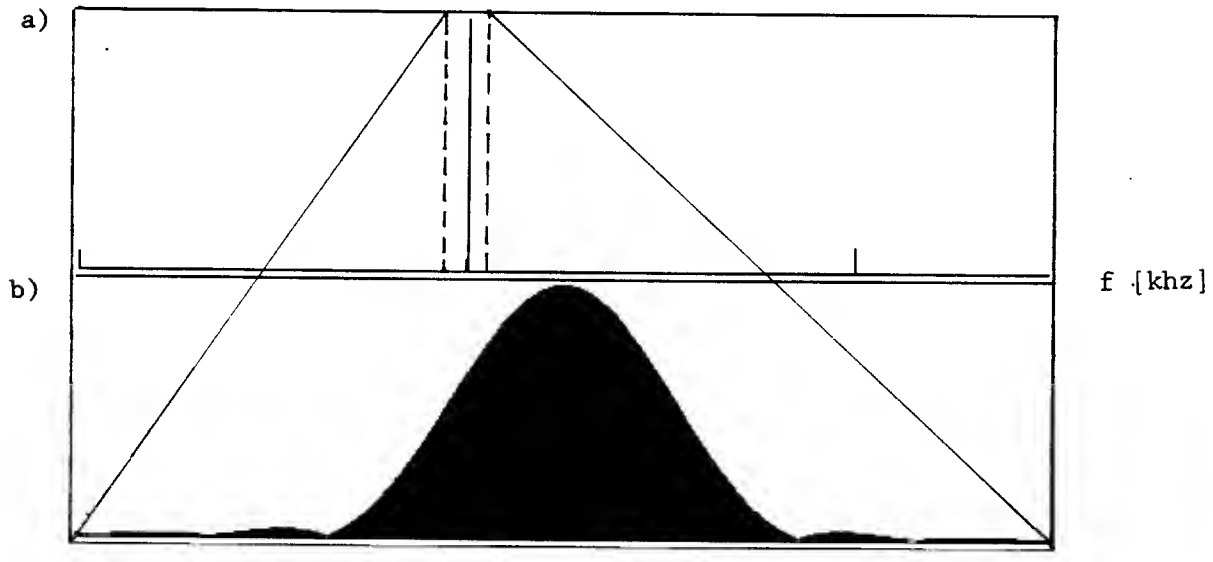

Figure 3.5: Power spectrum of a windowed signal. (a) The top graph illustrates the full spectrum on a linear scale. (b) The bottom graph shows a magnified version of (a) around the peak.

very little chance of accurately classifying the behavior of a system. The purpose of the program's peak-detection component is to sequentially check each point in the power spectrum to see which points possess the traits of a peak. Once it recognizes a peak, this module records the peak's height and location so the classification module can classify the recorded waveform's behavior.

3.2.1 Characteristics of a Peak

Each point in the power spectrum must pass three tests in the Peak Detection Module in order to qualify as a peak:[2] `tall-enough?`, `local-maxima?`, and `shape-of-a-peak?`.

The `tall-enough?` Criterion

In order to pass the `tall-enough?` criterion, the height of the potential peak must be sufficiently greater than the average height of the noise in the power spectrum.

Although windowing the input signal helps minimize spectral leakage and adjusting the range of the HP 5182a waveform recorder helps reduce the quantization error, there still exists a nontrivial amount of noise scattered throughout the power spectrum. Therefore, to prevent erroneous peak detection, it is important that the `tall-enough?` module verify that the average height of each potential peak is sufficiently greater than the average height of the low-level noise. To determine what is "sufficiently greater," the peak-detector module has a variable called `fuzz-threshold`. Any point with height less than this `fuzz-threshold` will not qualify as a peak point. The value of `fuzz-threshold` is a function of quantization error, arithmetic error and spectral leakage.

The `local-maxima?` Criterion

In addition to being taller than the noise floor, the potential peak must also be a local maximum. This means that the points to the right and to the left must be shorter than the potential peak. If, for example, the point to the right is taller, then the program

[2]The approach I've used for peak detection is very similar, though not based upon, the method used by Sreenivas and Rao[SR79] for pitch extraction.

discards the original point and adopts this right-hand point as a new peak candidate. It then runs the same tests on this new point to see if it meets all the criteria for a peak.

The shape-of-a-peak? Criterion

Besides being a local maximum and sufficiently tall, the potential peak must also have the shape of a peak. An ideal peak is characterized by a sharply monotonically-increasing slope and a sharply monotonically-falling slope.

The **sharp-enough?** module compares the ratio of the potential peak's height to that of a neighboring point. This ratio essentially gives the slope of the peak. If the slope is greater than a certain threshold value, then the peak is **sharp-enough?**. The advantage of using this method is its simplicity. The disadvantage is that the ratio is fairly sensitive to variations in the neighboring value. For example, it is possible to get a large ratio if the potential peak value is large *or* if the neighboring value is small. The first is desirable whereas the second is not. Yet the program avoids erroneous peak detection by imposing the **tall-enough?** condition, since the height of the potential peak must be greater than the height of the low-level noise. This prevents the peak-detector module from picking up extraneous short peaks merely because the neighboring value is smaller than usual.

Periodogram Averaging or Bartlett Estimator

For the many test cases I have run, the power spectrum was clean enough so the detector module could detect at least the salient peaks in the spectrum. However, in the event that the power spectrum, which is normally computed by taking a single FFT of the entire input signal, is too noisy and peak detection becomes too difficult, I have written an additional module capable of producing a cleaner power spectrum. Although computationally more expensive, this optional module can produce a clearer power spectrum using the periodogram averaging or Bartlett's estimator technique[OS75, AN83, SS75, RG75] as outlined below.

To get cleaner looking power spectra,

```
1) take signal     |---------------------- ... -----------------|
   of 16384 pts.
```

```
2) divide signal  |---1st chunk--|
   into overlapping      |---2nd chunk---|
   chunks of size              |---3rd chunk---|
   2000 points                         .
                                       .
                                       .
                                     etc.
3) FFT the chunks

4) Average the results.
```

More formally, in the periodogram averaging approach, a data sequence $x(n)$, $0 \leq n \leq N-1$, is divided into K segments of M samples each so that $N = KM$; i.e., we form the segments

$$x^{(i)}(n) = x(n + iM - M), \qquad 0 \leq n \leq M-1, \quad 1 \leq i \leq K \tag{3.2}$$

and compute the K periodograms

$$I^{(i)_m}(w) = \frac{1}{M}\left|\sum_{n=o}^{M-1} x^{(i)}(n)e^{-\jmath\omega n}\right|^2, \qquad 1 \leq i \leq K \tag{3.3}$$

The spectrum estimate is then defined as

$$B_{xx}(\omega) = \frac{1}{K}\sum_{i=1}^{K} I_M^i(\omega) \tag{3.4}$$

Choosing the optimal chunk size is important to ensuring a clean power spectrum. In general, it is not obvious how to choose the optimal chunk size. Increasing or decreasing the chunk size introduces tradeoffs which must be weighed. In general this choice may be guided by prior knowledge of the signal under consideration. Refer to [OS75, RG75, OS75, AN83]. For the purposes of this program, an overlap of 50% works best[OS75].

3.2.2 Finding the precise location of a peak

Once the program has identified a peak, it must record the peak's location so the classification component can determine its relationship to other peaks.

To compute the *Frequency* of a peak, the program uses Equation 3.5,

$$\left(\frac{Frequency}{Frequency\ Span}\right) = \left(\frac{Index}{Length\ of\ Power\ Spectrum}\right) \tag{3.5}$$

where $FrequencySpan$ and $LengthofPowerSpectrum$ are given, and $Index$ is computed by the peak-detector module. If each data point is in a one-to-one correspondence with each frequency point in the power spectrum, then equation 3.5 provides a simple and efficient way to determine the location of a peak. However, in some cases, there is far less than a one-to-one correspondence between points and frequencies. The forced negative-resistance oscillator exhibits most interesting behavior at frequencies from 50 khz to 500 khz. Since the power spectrum in this program can be represented by at most 8192 data points,[3] and the power spectrum must span 500 khz in order to reveal the interesting behavior, there is a $1 : (\frac{500000}{8192}) = 1 : 6.103515625$ correspondence between each data point and frequency. In other words, each data point in the power spectrum represents 6.103515625 hertz. As a result, the program might, in the worst case, be off by 6.103515625 hertz when it returns the location of a peak using equation 3.5.

To alleviate the problem, I have implemented an additional module that computes more precisely the location of peaks based on the initial, rough estimate.[4] The strategy is as follows.

First, for each peak, magnify the power spectrum around the peak using the chirp z-transform algorithm. See figure 3.6. The advantage of the chirp z-transform as it is implemented in the 6.003 software is that it allows the user to select the starting and ending frequencies, the incremental step size and the desired number of points for the magnified spectrum, unlike the fast Fourier transform procedure which does not afford such flexibility. In this program, I used 2048 data points to represent a magnified

[3]The HP5182a Waveform Recorder/Generator can capture a maximum of 16384 points of a single signal. Although the 6.003 FFT procedure outputs a spectrum of 16384 data points, half of these points provide redundant information since the power spectrum for the negative frequencies is merely a mirror image of the spectrum for the positive frequencies. Thus, after discarding the negative frequencies, we are left with $16384/2 = 8192$ data points in the power spectrum.

[4]Finding the precise location of a peak is extremely important for classification purposes because of the integral relationships between subharmonic frequencies. Section two of this chapter will discuss this issue in greater depth.

frequency span of 30 hertz. This means that each peak location can be off by at most $30/2048 = 0.0146484375$ hertz, a significant improvement over 6.103515625.

Second, take the tallest three points of the magnified spectrum, and fit them to a parabolic curve. Then use parabolic interpolation to find the center frequency of the peak.[5] In general, interpolation is necessary because the peak's center may lie *between* two data points, and the use of a parabola is effective because peaks, when magnified, have shapes similar to inverted parabolas. See figure 3.6.

More specifically, the equation for a parabola is:

$$Ax^2 + Bx + C = y \tag{3.6}$$

Fitting the three points, (x_1, y_1), (x_2, y_2), and (x_3, y_3), to a parabolic curve requires solving three simultaneous equations in three unknowns.

$$Ax_1^2 + Bx_1 + C = y_1 \tag{3.7}$$

$$Ax_2^2 + Bx_2 + C = y_2 \tag{3.8}$$

$$Ax_3^2 + Bx_3 + C = y_3 \tag{3.9}$$

Solving for A, B, and C, we get

$$B = \frac{(y_3 - y_1)(x_2^2 - x_1^2) - (y_2 - y_1)(x_3^2 - x_1^2)}{(x_3 - x_1)(x_2^2 - x_1^2) - (x_2 - x_1)(x_3^2 - x_1^2)} \tag{3.10}$$

$$A = \frac{(y_2 - y_1) - B(x_2 - x_1)}{(x_2^2 - x_1^2)} \tag{3.11}$$

$$C = y_1 - Ax_1^2 - Bx_1 \tag{3.12}$$

[5]This technique was suggested by Gerry Sussman.

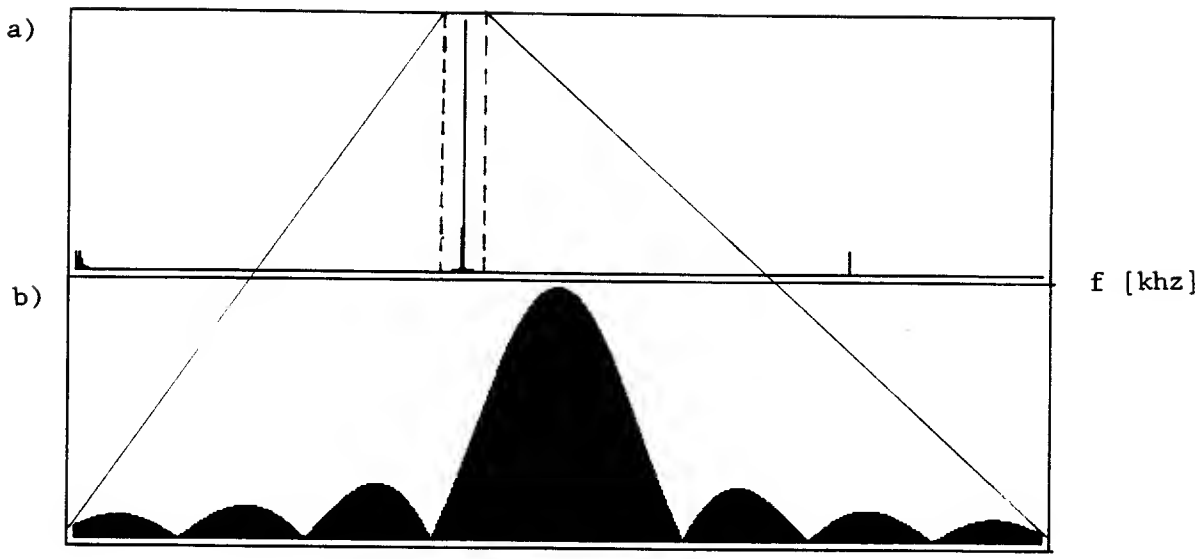

Figure 3.6: Magnification of Peak. (a) The top plot illustrates the full spectrum on a linear scale. (b) The bottom plot displays a magnified version of (a) around the peak.

Once the values for A, B, and C are known, then the center frequency can be readily computed. This center frequency is nothing more than the maximum value of the parabola obtained by taking the derivative with respect to x of equation 3.6.

$$2Ax + B = 0 \tag{3.13}$$

Solving for x,

$$Center\ Frequency = -\frac{B}{2A} \tag{3.14}$$

The third and final step is to take this center frequency of the peak and rescale it so it has meaning relative to the entire frequency span, not just the magnified span. Figure 3.6 and figure 3.7 illustrate the process.

Magnifying each peak and interpolating to find the center frequency proves very effective. The result is a peak location much more accurate than that obtained using equation 3.5. The tradeoff, however, is that the chirp z-transform is computationally expensive. If there are many peaks in the power spectrum, then determining the precise location of all the peaks becomes very slow.

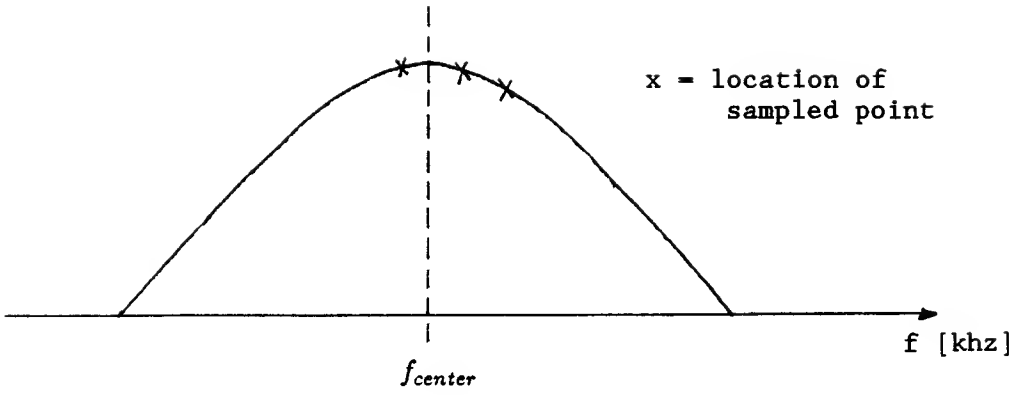

Figure 3.7: Parabola Fitting and Finding Center Frequency.

3.3 Classification Component

In the previous section, I describe the traits the peak-detection component looks for when identifying peaks and how it computes their locations in the power spectrum. Once the peak detector has gathered all the peaks and the frequencies at which they occur, it passes this list of peaks onto the classifier. In this section, I discuss the classification component. The purpose of this module is to determine if the response exhibits harmonic, subharmonic, quasiperiodic or chaotic behavior. I begin this section with a precise definition of the terms. Following the definitions is an explanation of each of the modules within the classification component.

3.3.1 Defining the Terms

When looking at the power spectrum, the program will use the following simple definitions to identify the system's behavior. Refer to figure 3.8 and figure 3.9 for examples.

- *Harmonic response* — peaks in the power spectrum may occur only at the drive frequency, f_{drive}, and integer multiples of the drive frequency.
- *Subharmonic response of order n* — peaks in the power spectrum may occur only at $\frac{1}{n} \cdot f_{drive}$ and integer multiples of $\frac{1}{n} \cdot f_{drive}$.
- *Quasiperiodic response of degree n* — peaks in the power spectrum may occur only at n fundamental frequencies and integer combinations of these fundamental

frequencies. For example, a quasiperiodic response of degree 2 may have peaks only at

$$\text{frequencies} = a \cdot f_1 + b \cdot f_2$$

where f_1 and f_2 are not rational multiples of each other and a and b are integers.[6]

- *Chaotic response* — the power spectrum has a lot of broadband noise in addition to some sharp peaks. The level of noise in the chaotic case is much greater than the level of noise for the harmonic, subharmonic, and quasiperiodic cases.

3.3.2 Chaos Module

The Classification Component of the program is made up of several modules, the first of which is the chaos module. Refer to Figure 3.10. The purpose of the chaos module is to identify chaotic behavior using evidence from the power spectrum.

As mentioned in the definition above, the power spectrum of a chaotic response is characterized by broadband noise in addition to some sharp peaks. In addition, the level of noise is much greater in the chaotic case than the low-level noise observed in the periodic and quasiperiodic responses. Figure 3.11 provides an example of a power spectrum for a chaotic response.

To recognize chaotic behavior in a power spectrum, I tried computing the average height of the spectrum and comparing this with the average height of a known, non-chaotic power spectrum. If the average height of this potentially chaotic response is significantly greater than that of a known, nonchaotic response, then I classified the system as chaotic. However, this proved to be an unreliable test for chaos.

An alternative strategy for discerning chaos in the power spectrum involves searching for sharp, distinct peaks in the spectrum. If the peak detector returns no sharply defined peaks, we may conclude that the spectrum contains only broadband noise and hence classify the behavior as `CHAOTIC`. However, the fact that a response has sharply defined peaks in its spectrum does not preclude it from being classified as chaotic. As indicated in the definition of chaos, it is perfectly possible for a chaotic response to have some

[6] The task of identifying quasiperiodicity is simplified slightly by the fact that the degree of a quasiperiodic response seldom exceeds 2. According to Thompson and Stewart[TS86], page 196, "A remarkable theorem in qualitative dynamics, proved recently by Ruelle, Takens, and Newhouse, suggests that quasiperiodicity of degree greater than 2 may be difficult to observe in nature."

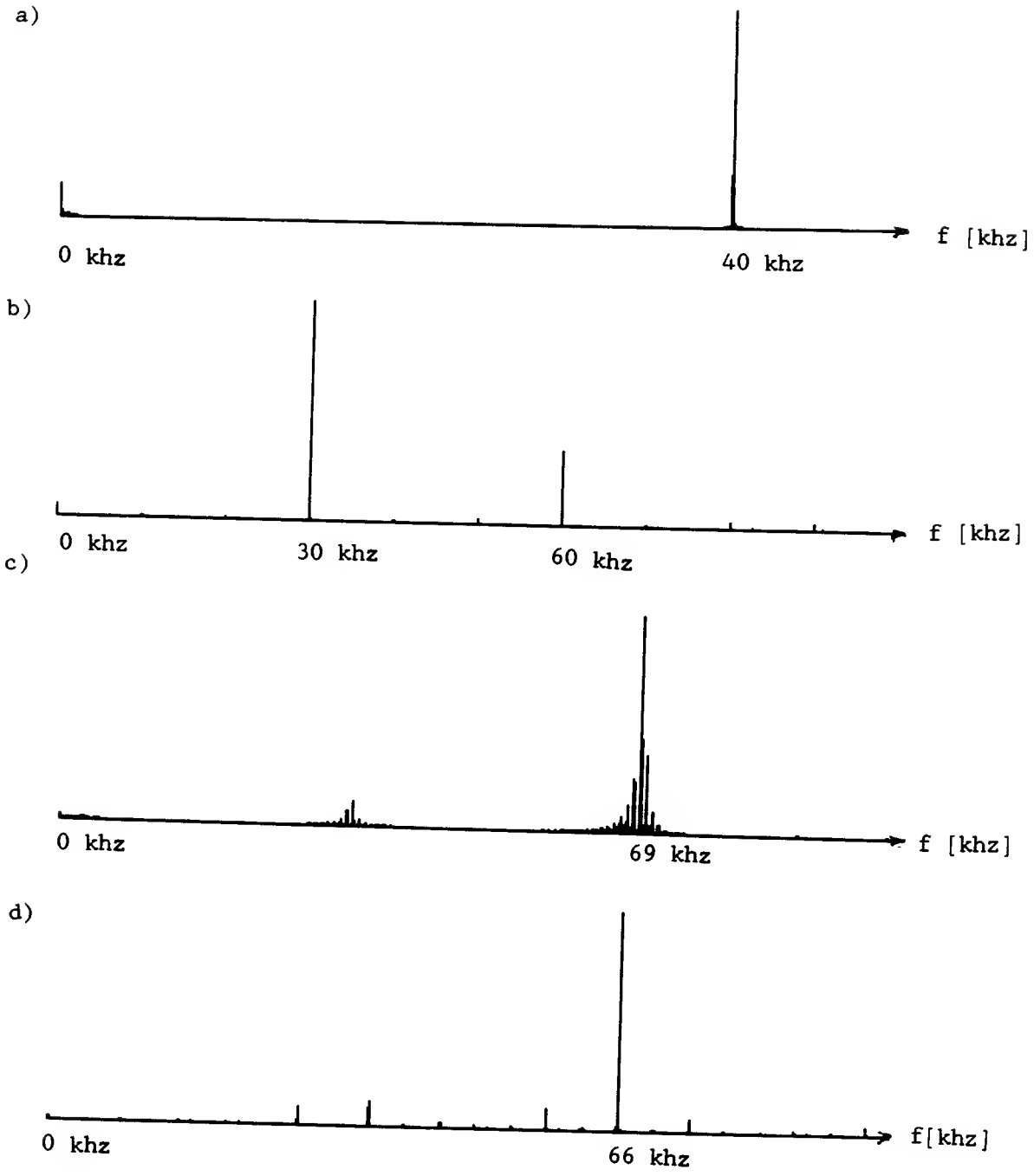

Figure 3.8: Examples of power spectra on a linear scale: (a) a harmonic response with f_{drive} = 40000 hz; (b) a second-order subharmonic response with f_{drive} = 60000 hz; (c) a quasiperiodic response with f_{drive} = 69000 hz; and (d) a chaotic response with f_{drive} = 66000 hz.

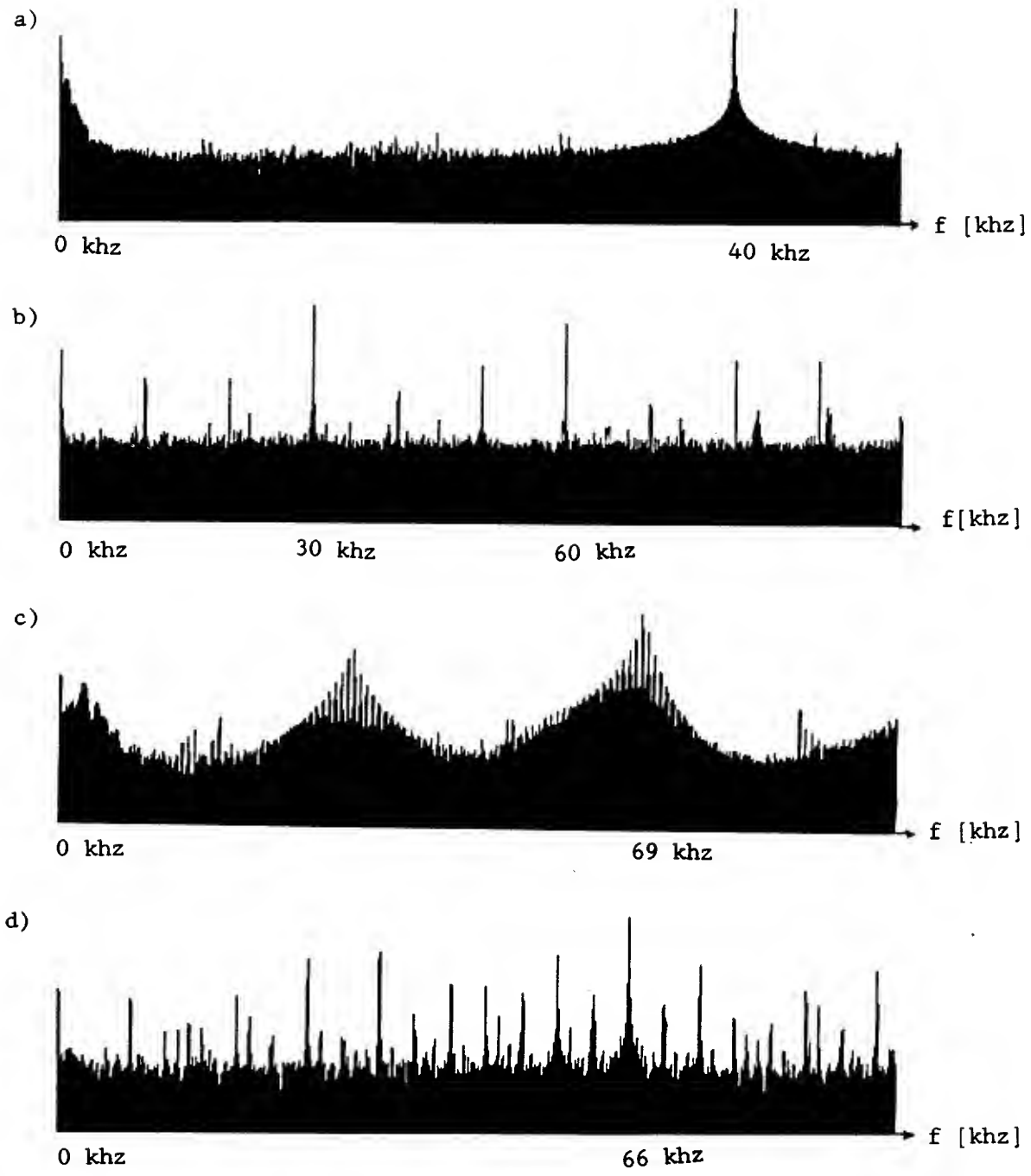

Figure 3.9: Examples of power spectra on a logarithmic scale: (a) a harmonic response with f_{drive} = 40000 hz; (b) a second-order subharmonic response with f_{drive} = 60000 hz; (c) a quasiperiodic response with f_{drive} = 69000 hz; and (d) a chaotic response with f_{drive} = 66000 hz.

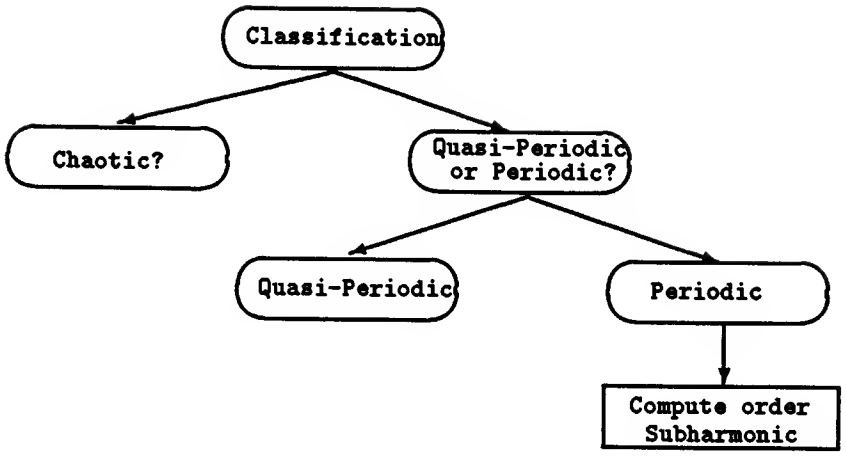

Figure 3.10: Components of the Power Spectrum Classification Module.

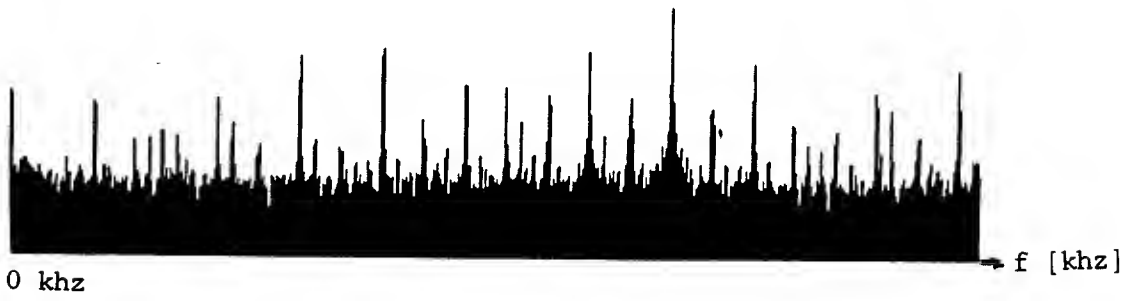

Figure 3.11: Power spectrum of a chaotic response.

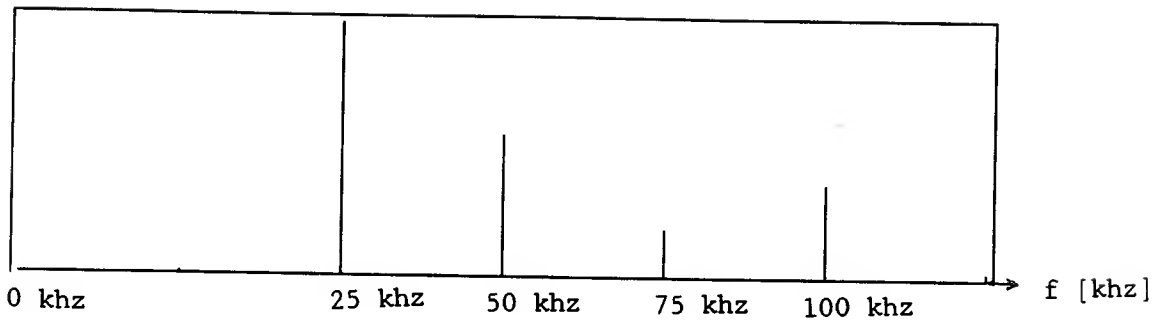

Figure 3.12: Power spectrum of a second-order subharmonic response with $f_{drive} = 50$ khz.

sharp peaks. I have found that the power spectrum does not easily provide conclusive information regarding the chaotic versus non-chaotic nature of a signal.[7] In the next chapter, I present the autocorrelation technique as a promising alternative approach to distinguishing between quasiperiodicity and chaos.

Referring back to figure 3.10 again, we find that if the program has concluded the response is not chaotic, its next step is to distinguish between quasiperiodic and periodic behavior. Before discussing how the program makes this distinction, I wish to first explain how the program determines the order subharmonic of a periodic response. An understanding of how the program computes subharmonics will aid in the understanding of how it distinguishes between periodicity and quasiperiodicity.

3.3.3 Subharmonic Module

Assuming we already know the response is periodic, the next goal is to determine its order. Figure 3.12 shows the power spectrum of a second-order subharmonic response whose drive frequency is 50 khz.

We recognize this response as second-order subharmonic because peaks occur only at integer multiples of:

[7] As it turns out, the problem of distinguishing between quasiperiodic and chaotic behavior is fairly difficult[UA81, KKC].

$$\frac{1}{n} \cdot f_{drive} \quad where \ n = 2 \ and \ f_{drive} = 50 \ khz \tag{3.15}$$

at frequencies

$$25 \ khz = \tfrac{1}{2} \cdot f_{drive},$$
$$50 \ khz = \tfrac{2}{2} \cdot f_{drive},$$
$$75 \ khz = \tfrac{3}{2} \cdot f_{drive}, \quad \text{and}$$
$$100 \ khz = \tfrac{4}{2} \cdot f_{drive}$$

Information contained in the first 50 khz of the power spectrum is sufficient for us to classify the system's response as second-order subharmonic. However, the information contained in the second segment from 51 khz to 100 khz also provides valuable supporting evidence for our hypothesis. In particular, the peak at 75 khz confirms the information contained in the 25 khz peak, while the peak at 100 khz confirms the information contained in the 50 khz peak. Scientists, especially when confronted with a noisy power spectrum, often search for this kind of redundant evidence to confirm or disconfirm their hypothesis. To take advantage of this duplicate information, I have chosen a multi-set representation for the list of peaks. This multi-set representation lumps together similar peaks much as a scientist looking at a power spectrum would. For the power spectrum shown in figure 3.12, the list of peaks is

```
((25000 . 10.4206) (50000 . 5.954) (75000 . 1.9) (100000 . 3.721))
    ^         ^
    |         |_____ height of peak
    |
location of peak
```

where the first element of each set indicates the location of the peak (in hertz) and the second element indicates the height of the peak. This list of peaks will get transformed into the multi-set

```
((25000  .  2) (50000  .  2)).
   ^        ^
   |        |
normalized  number of occurrences
location of
  peak
```

with the first element of each set specifying the normalized[8] location of analogous peaks and the second element specifying the number of occurrences of such analogous peaks. Thus, in this case, the (25000 . 2) entry indicates that there were two occurrences of the 25000 hertz "type" peak in the power spectrum, one at 25000 hertz and one at 75000 hertz. Similarly, there were two occurrences of the 50000 hertz "type" peak, one at 50000 hertz and one at 100000 hertz.

Once the program has computed this multi-set of remainders, it then transforms this multi-set of remainders into a multi-set of hypotheses by dividing the drive frequency by each normalized location. Upon performing this division, we get a new multi-set of hypotheses.

```
((2  .  2) (1  .  2))
  ^     ^
  |     |___ number of occurrences
  |
hypothesized
order subharmonic
```

To compute the final order of the system's response, the program takes the least common multiple of all the hypothesized orders. In this example, (lcm 2 1) = 2, indicating a second order subharmonic response, which is correct.

If the peak detector fails to return the precise location of the peaks, but instead returns the following list:

[8]The normalized location is computed by taking the (*peak location*) mod (*drive frequency*).

```
((25000.003 . 2.8) (50000.012 . 10.206) (75000.001 . 3.1) (100000 . 6.229))
     ^         ^
     |         |_____ height of peak
     |
location of peak
```

the program can still take advantage of the multi-set representation. This is true because the program tries to determine what the true location of the peak *should* have been. More specifically, if the computed location of the peak were 25000.003, but the true location is 25000, then the program tries to justify renaming the 25000.003 peak to a 25000 peak. If the computed peak at 25000.003 is `close-enough?` to the ideal peak location at 25000, then the program goes ahead and renames the 25000.003 peak to a 25000 peak. Computing the `close-enough?` threshold level is easy. Recall from section 3.2.2, we know how to determine the maximum deviation of a computed peak location from its true location based upon the frequency span and the number of points in the power spectrum. Thus, if a computed location is off from the ideal location by less than this maximum deviation, then the difference in distance between the ideal location and the computed location could logically be explained by the coarse granularity of the power spectrum.

In the event that the peak detector erroneously picks up a stray spike, the correct classification of the system's response becomes very difficult. The multi-set representation, however, helps the program to distinguish between true peak and stray spike.[9]

For instance, if the peak-detector module accidentally detected an unusually tall stray spike at 847.457627119 hertz, then we might get the following for the list of peaks:

```
((847.457627119 . 1.04) (25000 . 9) (50000 . 4) (75000 . 2.3) (100000 . 3)),
```

[9]Windowing the input to reduce spectral leakage, selecting the appropriate input range on the HP5182a waveform recorder and utilizing Bartlett's periodogram averaging technique discussed above *reduce* the low-level noise in the power spectrum which minimizes the probability of erroneous peak detection by the Peak Detector Module. But, the multi-set representation helps the program recognize stray peaks after the Peak Detector has already identified these strays as peaks.

with a multi-set of remainders of:

```
((847.457627119  .  1) (25000  .  2) (50000  .  2)),
```

and a multi-set of hypotheses of:

```
((59  .  1) (2  .  2) (1  .  2))
```

Taking the least common multiple of the 59, 2 and 1, we get 118. At this point the program is smart enough to recognize that a 118th order subharmonic response is 1) nearly impossible to measure because of the imprecisions of the instruments, and 2) is highly unlikely even if the instruments were extremely precise. Instead of returning a conclusion of `(SUBHARMONIC 118)`, it goes back to the multi-set of remainders and notices that there was only a single occurrence of `(847.457627119  .  1.04)`. Since there was only a single occurrence of the 847.457627119 "type" peak, this suggests that the 847.457627119 peak may be a stray. (Perhaps the peak-detector module erroneously picked it up.) The program then knows to toss out this single, stray peak and recomputes the order until the answer seems reasonable. In this case, throwing out the `(847.457627119  .  1)` term from the multi-set of remainders allows the program to recognize the system's response as second-order subharmonic, as it should.

Quasiperiodic-or-Periodic Module

Now that we know how the subharmonic module classifies the various order subharmonic responses, we can begin to discuss how it distinguishes between quasiperiodicity and periodicity.

Recall from section 3.3.3 that if there was a stray peak in the power spectrum, which caused the computed order to be extremely high, the program was smart enough to throw out this peak and recompute the order until it arrived at a reasonable order. For periodic responses there should be no more than a few of these stray peaks. In the section on the subharmonic module we saw how nicely the 25 khz peak got lumped with the 75 khz peak and how the 50 khz peak got lumped with the 100 khz peak when we normalized with the drive frequency of 50 khz, leaving no stray peaks.

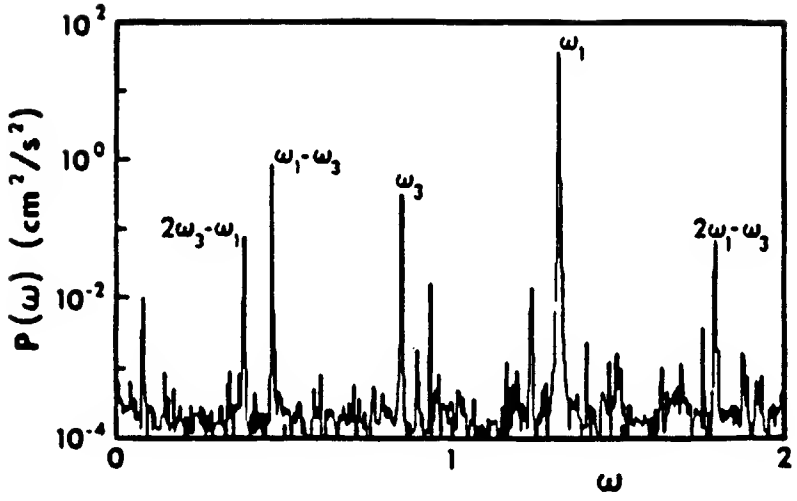

Figure 3.13: Power spectrum of a quasiperiodic response with two fundamental frequencies at ω_3 and ω_1.

The power spectrum of a quasiperiodic waveform, with sharp peaks at multiple fundamental frequencies and integer combinations of these fundamental frequencies, will be much more likely to have a greater number of single occurrence peaks than a periodic response. Referring to the quasiperiodic power spectrum in figure 3.13[TS86], we see peaks at the two fundamental frequencies, ω_1 and ω_3, along with peaks at some of their integer combinations, $2\omega_3 - \omega_1, \omega_1 - \omega_3$, and $2\omega_1 - \omega_3$. Normalizing all the peaks at $2\omega_3 - \omega_1$, $\omega_1 - \omega_3, \omega_3, 2\omega_1 - \omega_3$ by the drive frequency yields a different answer almost every time because each peak is not an integer multiple of another peak. Thus, it becomes much more difficult to lump together analogous peaks as we could in the periodic case when peaks tended to be multiples of another rather than sums and differences of one another.

To distinguish between quasiperiodicity and periodicity, the quasiperiodic-or-periodic module notices how many single occurrence peaks occur in the multi-set of remainders. If more than a few single occurrence peaks exist in a spectrum,[10] this suggests the system's response may be quasiperiodic.[11]

[10] For the purposes of this implementation of the program, "a few" means two single occurrence peaks. This is a fairly tight constraint. However, based upon the examples seen so far, the power spectra of periodic signals tend to have sharp peaks only at integer multiples of each other, producing very few single occurrence peaks in the multi-set.

[11] Actually, it would be nice if we could easily group together analogous peaks in the quasiperiodic spectrum just as we can in a periodic spectrum. For example, the peaks at $2\omega_3 - \omega_1, \omega_1 - \omega_3, \omega_3, \omega_1$, and

If the `quasiperiodic-or-periodic` module deduces that the power spectrum indicates quasiperiodic behavior, the program returns `(QUASIPERIODIC)`; otherwise, it proceeds to the subharmonic module and computes the order subharmonic as described in the previous section.

$2\omega_1 - \omega_3$ should all be lumped together since they are all integer combinations of the drive frequency, ω_1, and the other fundamental frequency, ω_3. The problem is that in order to recognize the analogous peaks, we must first be able to recognize all the fundamental peaks, which proves difficult. Even with the two fundamental frequencies known, there may be too many combinations to detect.

Chapter 4

The Time-signal Analyzer

In the previous chapter, we discussed how the power-spectrum data can be used to classify a system's response. In most cases spectral analysis can accurately categorize the behavior of a waveform. However, as mentioned in chapter 2, the power spectrum provides only one perspective of the system's behavior. When the noise level in the spectrum increases, peak detection and behavior classification become more difficult, thereby increasing the likelihood of erroneous classification. To supplement the information provided by the power-spectrum data, the program also looks at the time-signal data. The purpose of this chapter is to present the time-signal analyzer, which identifies the circuit's behavior using time signal information.

4.1 Time-signal definitions

Before discussing the details of implementation, I first present the definitions of harmonic, subharmonic, quasiperiodic, and chaotic from a time-domain perspective.[1] These are the definitions used by the time-signal analyzer to categorize the behavior of a signal.

- *Harmonic response* — a periodic response with the same frequency as the driving frequency. $T_{drive} = T_{response}$, where $T = period$.
- *Subharmonic response of order n* — a periodic response whose frequency is some fraction of the drive frequency. Or $T_{response} = n \cdot T_{drive}$, where $T = period$.

[1] The previous definitions given in chapter 3 were from a power spectrum perspective.

- *Quasiperiodic response* — a response that is the sum of a couple of periodic signals, whose periods are not rational multiples of each other. The signal is *not* periodic, but rather, *almost* periodic.
- *Chaotic response* — a response that cannot be simply be characterized. The time signal does not repeat itself in a finite amount of time.

4.2 Basic strategy of the time-signal analyzer

This section describes the strategy used by the time-signal analyzer to identify the behavior of a recorded waveform.

4.2.1 Strobe technique to determine order subharmonic

The time-signal analyzer first assumes the recorded waveform is periodic. It then determines the order subharmonic by strobing the signal at integer multiples of the drive period and recording the value of each strobe. Since the program knows the drive frequency,[2] it can compute the drive period using the relation $T_{drive} = 1/f_{drive}$. If all strobed values are within a given epsilon of each other, then the signal is said to be harmonic, since it repeats itself at intervals of a period. If every other strobe point has the same value, then the waveform is second-order subharmonic. If every third strobe point has the same value, then the waveform is third-order subharmonic, etc. Figure 4.1 through figure 4.4 illustrate the notion of strobing the signal to determine order subharmonic. In each of the figures, the top graph displays the forcing sinusoid, and the bottom graph shows the response.

In the actual implementation, I examine ten values spaced at multiples of the drive period. If each of the later nine points are within a small distance, epsilon, of the first point, then the signal is periodic, more specifically harmonic. If not, the program looks at ten values[3] spaced at twice the drive period. Once again if the later nine points are all

[2] Recall the inputs to the entire analyzer program are the ranges of frequency and voltage over which to run the analysis.

[3] For greater assurance of accuracy, you can increase the number of strobed points. Increasing the number of strobed values from ten to say fifty means that fifty points must be "close enough" to one another as opposed to ten. This is a much tighter constraint. However, if all fifty points are "close enough," then you can be more certain that the deduced subharmonic order is correct.

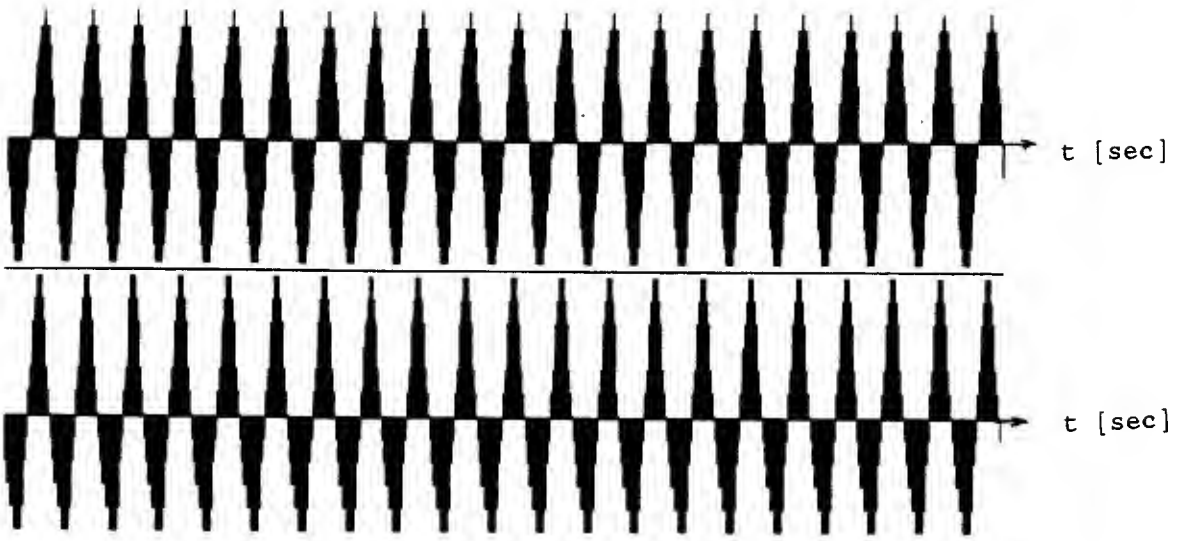

Figure 4.1: Strobing a harmonic response. Notice how the response waveform repeats itself at every strobe.

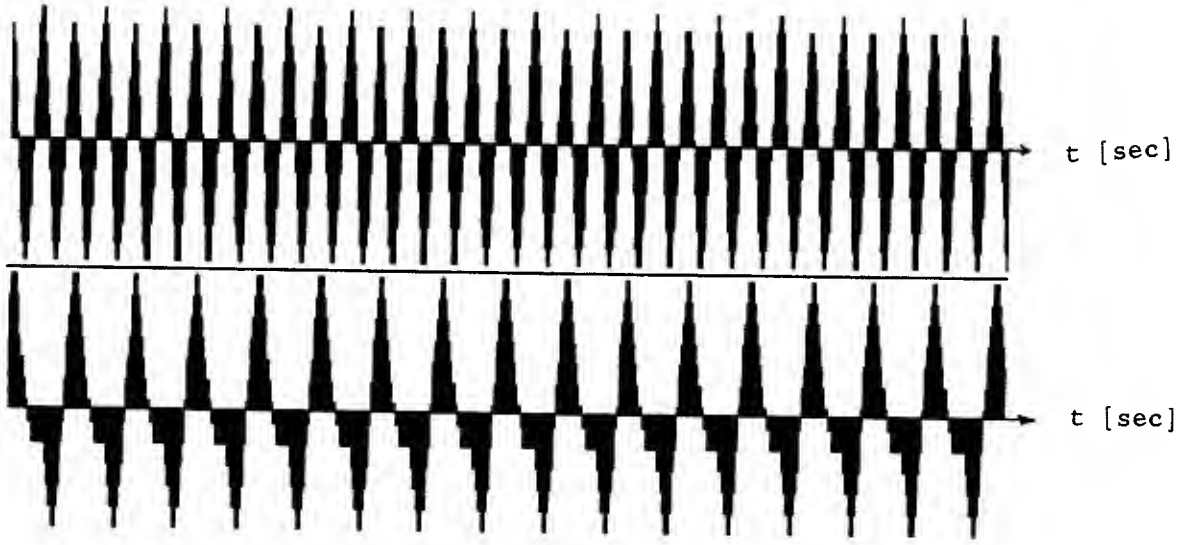

Figure 4.2: Strobing a second-order subharmonic response. Notice how the response waveform repeats itself once for every two times the driving sinusoid repeats itself.

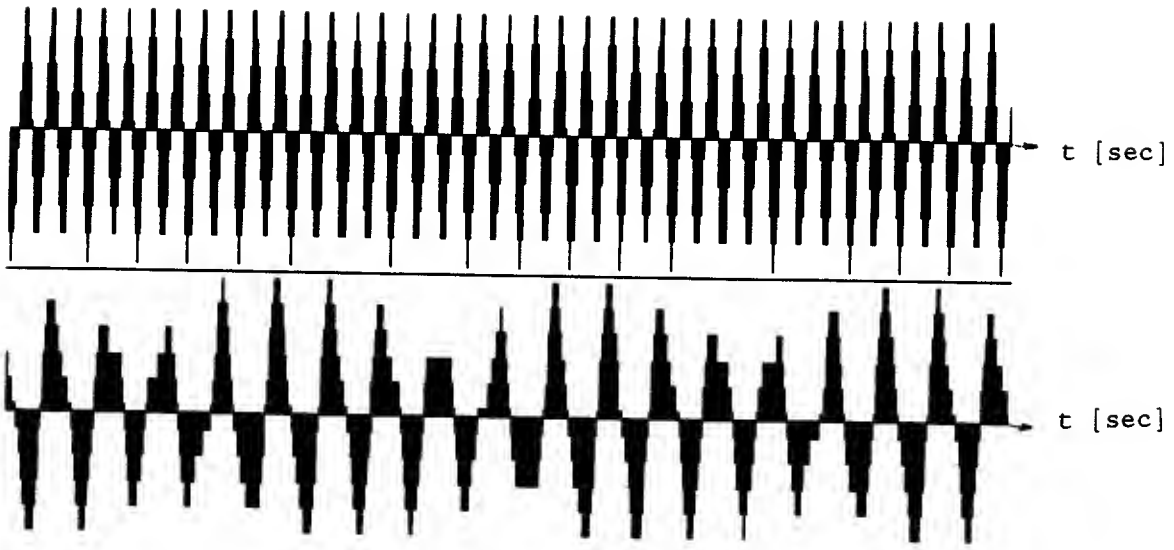

Figure 4.3: Strobing a quasiperiodic response.

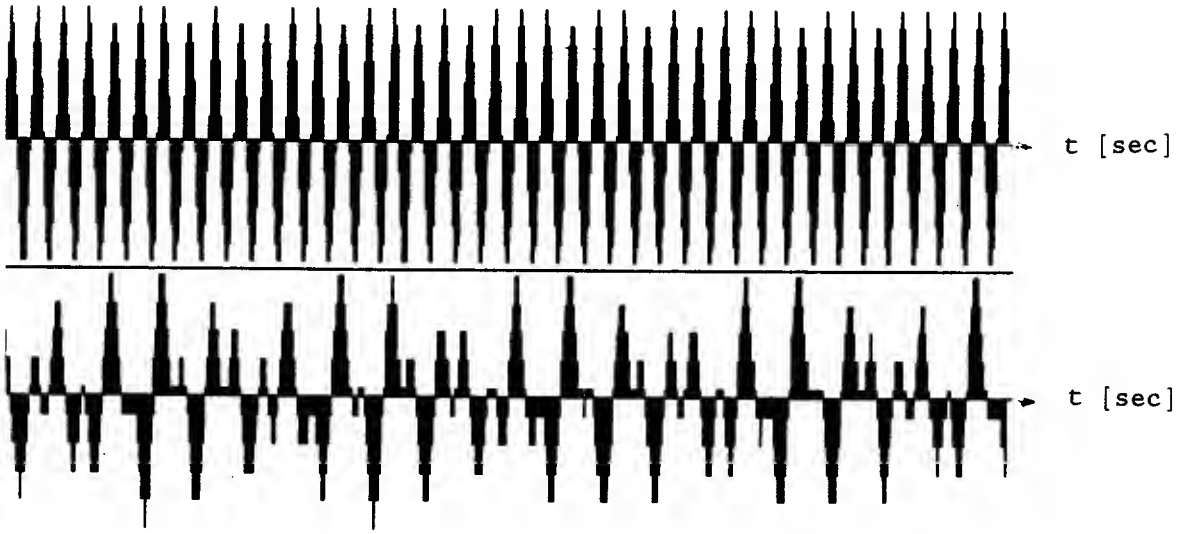

Figure 4.4: Strobing a chaotic response. Notice how the response never repeats itself, even with an infinite number of strobes.

within a short distance of the first point, then the program calls this signal subharmonic of order two, and so on up to some large order subharmonic, like thirty-three.[4] If the order subharmonic is greater than thirty-three, then the behavior of the response is probably quasiperiodic or chaotic, and the program classifies it as such.

To be truly precise, the time-signal analyzer should probably compare each strobed value with every other strobed value, not just the first one. In my implementation, the determination of the order subharmonic is highly dependent upon the value of the first strobed point. If for some reason the value of this first point has a large variance in comparison to the other nine points, then the analysis returned by the time-signal analyzer may be inaccurate. However, the advantage of this implementation is that it is simpler and more efficient. Moreover, it is unclear how much certainty we gain by checking extra point values. For the examples I have run, there does not seem to be a sacrifice in the accuracy of the conclusions resulting from comparisons with the later nine points with the first point only. If, however, this does prove to be a problem, we can easily modify the time-signal analyzer so it performs a point-by-point checking.

Finally we must address the issue of how the program computes epsilon. Recall that epsilon is the threshold value below which two strobed points are considered close enough and above which two strobed points are *not* close enough. The time-signal analyzer computes the epsilon value by taking a certain fraction of the signal span. More specifically, the program computes the difference between the maximum and minimum values in the time signal. This is the *span* of the signal. It then multiplies the span by a certain fraction, in this case 0.10. What this means is if a subsequent strobed point varies from the first strobed point by no more than 10% of the *span*, then the two points are essentially close enough, and the signal has a chance of repeating itself in a periodic or quasiperiodic manner.

[4] There is no theoretical justification for the selection of subharmonic order thirty-three as the threshold between periodicity and non-periodicity. The only real reason for this selection is that subharmonics higher than thirty-three are very difficult to detect due to such factors as imprecision in the measurement instruments and rounding in the signal-processing package.

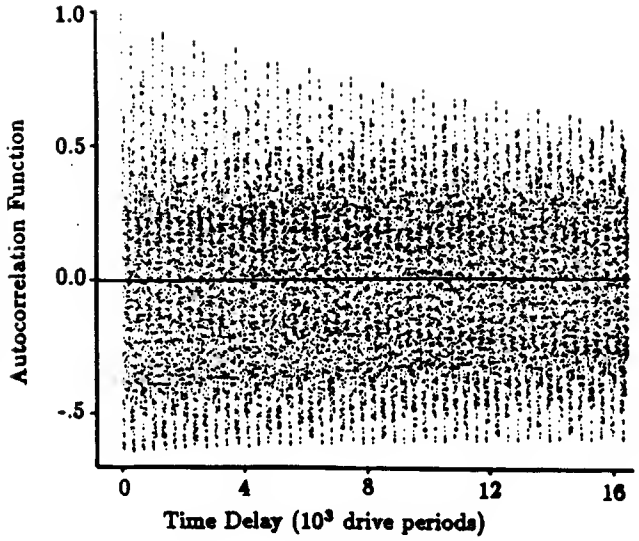

Figure 4.5: Autocorrelation of a Quasiperiodic Response.

4.2.2 Autocorrelation technique to distinguish between quasiperiodicity and chaos

The previous section described how the time-signal analyzer computes the order subharmonic of a periodic response. But how does the analyzer recognize a chaotic or quasiperiodic response? One possible technique for distinguishing between a quasiperiodic and a chaotic response is to compute the autocorrelation of the recorded waveform.[5] If the signal remains fairly highly correlated with itself through time, it is periodic or quasiperiodic. See Figure 4.5[Lin88]. If, on the other hand, the correlation values are very low as shown in figure 4.6[Lin88], then the signal is chaotic.

An autocorrelation value of zero is said to be *uncorrelated*, while a large autocorrelation value indicates high correlation. When we look at figure 4.6, it makes sense that the chaotic signal, which never repeats itself, has low correlation values for large time delays and high correlation when the time delay equals zero, since the signal correlates exactly with itself, but not very well with the shifted versions of itself. The quasiperiodic signal has the property that it is "almost periodic"; that is, the signal almost repeats itself. Therefore, as shown in figure 4.5, its autocorrelation function has fairly large values in

[5]This strategy of using the autocorrelation to distinguish between quasiperiodicity and chaos was suggested during a discussion with Professor Paul Linsay of the MIT Physics Department.

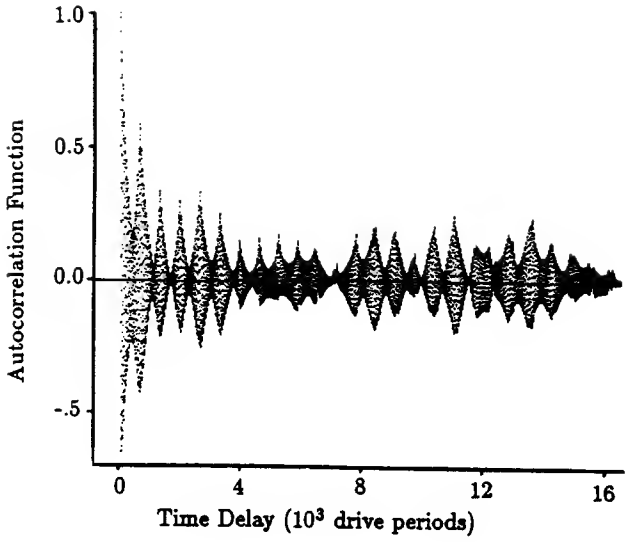

Figure 4.6: Autocorrelation of a Chaotic Response.

comparison to the the chaotic case, but as the time delay grows larger, the amount of correlation also decreases.

The notion of using the autocorrelation of a signal to distinguish between quasiperiodic and chaotic responses looks promising. However, I have not yet implemented this component yet. Paul Linsay[Lin88], in his studies of quasiperiodicity and chaos, has used the autocorrelation technique extensively to distinguish between the two types of behavior and finds this method highly effective. The implementation of the autocorrelation component of the time-signal analyzer has been left for future work.

4.3 The Resolver

What happens if the conclusion produced by the power-spectrum analyzer disagrees with the conclusion of the time-signal analyzer? In such cases, the resolver module must resolve the conflict. To decide which one of the two conclusions to accept, the resolver will look at *contextual* information. If, for example, the power-spectrum analyzer identified the behavior as `CHAOTIC`, and the time-signal analyzer classified the behavior as `SUBHARMONIC ORDER 2`, the resolver will look at the neighboring points to the east and to the south. If both are `SUBHARMONIC ORDER 2`, then the resolver will assume that the power spectrum was extraordinarily noisy, and it will reject the `CHAOTIC` analysis and

accept the SUBHARMONIC ORDER 2 analysis. This strategy is based upon the assumption that within a given local region, the behavior of a system will tend to behave in a similar manner. The validity of this assumption depends upon the step size or granularity with which we have chosen to run the program and the rapidity with which the circuit changes behavior. Contextual information provides just one method of resolving conflicting analyses. The final chapter of this thesis suggests others.

Chapter 5

The High-level interpreter

In the previous two chapters of this thesis, I discussed the classification module. Recall that the primary purpose of the classification module is to characterize the behavior of the nonlinear system for every point in the two-dimensional parameter space. In this chapter, I present the high-level interpreter. Its purpose is to take the detailed parameter-space graph produced by the classification module and to provide a high-level, textual explanation describing any interesting, qualitative properties of the circuit's behavior. For example in chapter 1, we saw that the high-level interpreter generated the following textual explanation given the parameter-space graph shown in figure 5.1.

HIGH-LEVEL INTERPRETATION OF PARAMETER-SPACE DIAGRAM

```
The parameter space diagram characterizes the behavior of the
forced negative-resistance oscillator as the frequency and amplitude
of the driving sinusoid are varied from 40000 to 70000 hz and from
1.5 to 6.5 volts respectively.  For amplitudes between 1.5 and 6.5
volts and frequencies between 40000 and 56000 hertz, the system ex-
hibits predominantly first order subharmonic behavior, while for
amplitudes between 1.5 and 5.75 volts and frequencies between 41000
and 68000 hertz, the system exhibits predominantly chaotic behavior,
while for amplitudes between 1.5 and 6.5 and frequencies between
55000 and 70000 hertz the system exhibits predominantly second order
subharmonic behavior.  Aside from these large areas of uniform behavior,
the system's response passes briefly through subharmonics of order
3, 4, 5, 7 and 9 at isolated points throughout the parameter-space
graph.
```

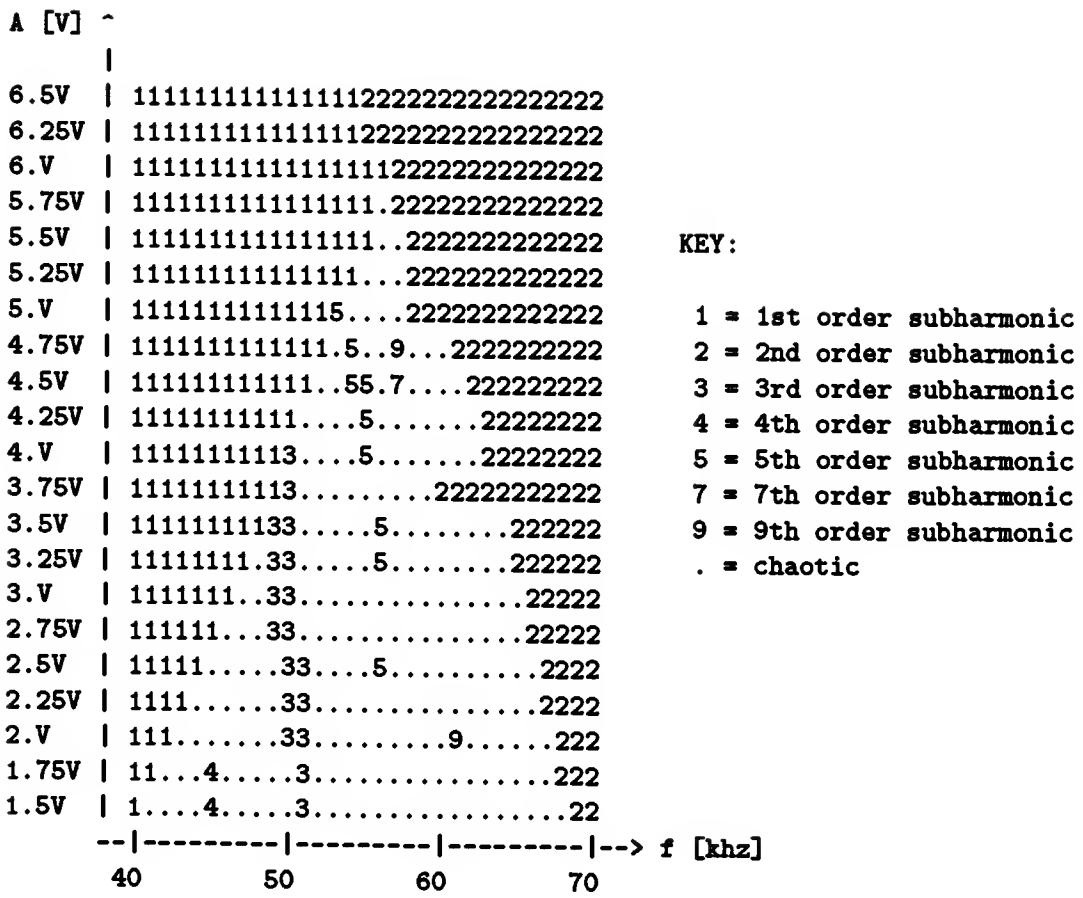

Figure 5.1: Sample parameter-space diagram.

```
The large region of first order subharmonic behavior is bounded
by the bifurcation curves 1-2 and 1-C, where C is a region consisting
of primarily chaotic behavior, but littered with sporadic occurrences
of other kinds of behavior.  Similarly, the large region of second
order subharmonic behavior is bounded by the bifurcation curves 2-1
and 2-C, while the region of chaos is bounded by C-1 and C-2 bifur-
cation curves.

Finally, based upon the data gathered so far, the forced neg-
ative-resistance oscillator shows no clear signs of period-doubling
bifurcations as described by Feigenbaum and others.
```

In order to provide such a high-level description of the circuit's behavior, the interpreter must identify large regions of uniform behavior as well as the small regions of sporadic behavior. It must notice the types of boundaries which border each region, and finally it must recognize such well-studied patterns as period-doubling routes to chaos and period-adding routes to chaos. How the program accomplishes all these tasks is the topic of this chapter.

The first section in this chapter discusses how the high-level interpreter scans the parameter space to produce the first paragraph of the textual explanation, which describes the different regions of behavior. The second section of this chapter explains how the interpreter uses computer vision techniques to detect the edges of a region and to locate the bifurcation curves that border the regions of uniform behavior. The third describes how the interpreter uses knowledge about general nonlinear dynamics to identify such typical patterns such as period-doubling cascades and period-adding cascades in the parameter space. The fourth section talks discusses the symbolic description returned by the analyzer program and used as input to other programs.

5.1 Growing regions of uniform behavior

This section describes how the high-level interpreter identifies the large and small areas of uniform behavior in the parameter space. By utilizing the computer vision technique of *growing* regions of uniform behavior[Hor86], the interpreter can locate the regions of qualitatively distinct behavior. The strategy for growing such regions is as follows.

The program begins with any point in the parameter space, say the origin. This becomes the *seed* point. It then tries to grow the seed in all possible directions: north, south, east and west. If, for instance, the behavior of the point to the north is the same as the behavior of the seed point, then the area of this region expands to include the northern point. In other words, any point contiguous to the seed point and having the same behavior as the seed point becomes incorporated into the growing region. After this northern point has been `CLAIMED` by a given region, it can no longer be `CLAIMED` by another region. Instead, it will serve as a new seed point for further growth of the region. When the region cannot grow any further in any direction, the program systematically scans the parameter space point by point for the first `UNCLAIMED` point to serve as a new seed point for a new region of behavior. Finally, when all the points in the parameter space are `CLAIMED`, we have a parameter space in which all contiguous points having the same behavior are grouped together as desired. See Figure 5.2. Given this grouping of points, the high-level interpreter can easily generate statements similar to those shown in the first paragraph of the sample textual explanation. It can describe the large regions of uniform behavior, draw attention to isolated regions of sporadic behavior, make note of general changes in behavior as an input parameter is varied, and comment upon the frequencies and voltages at which these regions of behavior occur.

5.2 Recognizing boundary types of a region

Generating the textual explanation for the second paragraph, however, is not as straightforward. Rather than requiring only local information about the behavior of neighboring points, the program now needs more global information. For instance, it must be able to look at the parameter-space diagram in figure 5.1 and recognize that the large region of first-order subharmonics is bounded by two kinds of borders: a 1-C border, and a 1-2 border, while the region of second-order subharmonics is bounded by a 2-1 bifurcation curve and a 2-C bifurcation curve. The high-level interpreter must also recognize that the streaks of 3rd, 4th, 5th, and 9th-order subharmonics in the chaotic region do not change the predominance of chaotic behavior in the middle frequencies. In other words, the behavior in the middle region should be viewed as mostly chaotic with sporadic

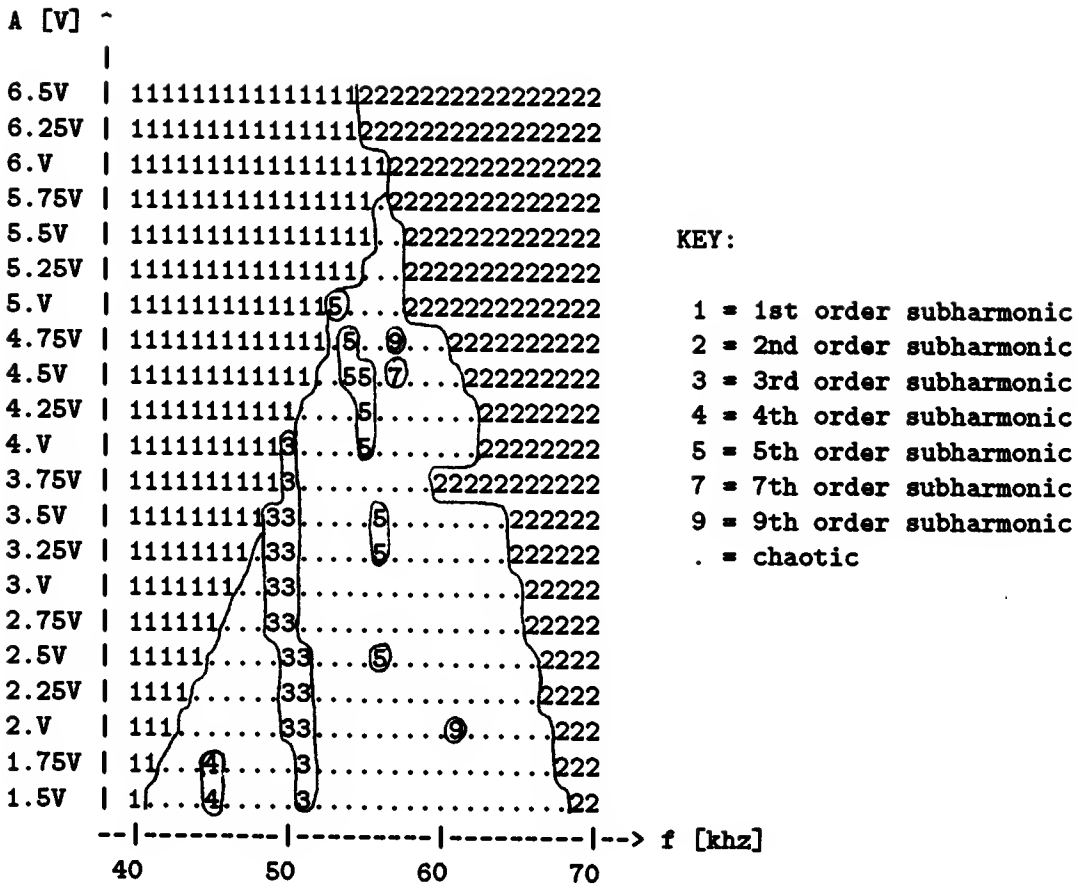

Figure 5.2: Lumped regions of uniform behavior.

occurrences of other types of behavior, rather than as ten separate regions of behavior.

Figure 5.3 illustrates the overall strategy of the High-Level Interpreter in recognizing the boundary types of a region. In general the task may be decomposed into two main parts. The first, which is illustrated in the first column of blocks in Figure 5.3, is to identify the large regions of uniform behavior. The second, which is illustrated in the second column of blocks in Figure 5.3, is to detect the bifurcation curves bordering each region of uniform behavior.

5.2.1 Identifying large regions of uniform behavior

In cases where the parameter-space diagram is not littered with small, isolated occurrences of sporadic behavior, recognizing the regions of uniform behavior is easy. The program need only grow points of similar behavior using the algorithm described in the previous section. If, however, the parameter space is littered with brief occurrences of sporadic behavior or "blemishes," the interpreter attempts to remove these local blemishes, since they distract from the overall picture. A blemish is defined as a small region in the parameter space having an area less than a certain fraction of the total parameter space area.[1] To remove a blemish, the interpreter checks to see if the neighbor to the north belongs to a large region of uniform behavior. If it does, the interpreter will copy the behavior of the northern point into the behavior of the current point, thereby erasing the blemish. If the northern point does not belong to a large region of uniform behavior, the interpreter will check to the south, then to the east and then to the west.[2] When all the blemishes in the entire parameter space have been removed, the program re-grows the regions of uniform behavior. After the second growth, there should be fewer distinct regions in the parameter space, but each region should have a greater area or a greater number of points in it. Figure 5.4 illustrates the result of removing the blemishes from the parameter space in Figure 5.1. Notice how the program can now easily recognize the three large regions of uniform behavior: first-order subharmonic, second-order

[1] Currently the threshold area is set at 3% of the total area.

[2] The problem of blemishes in the parameter-space graphs is similar to Horn's problem of *salt-and-pepper noise* in a picture. To remedy the problem of "noise," we both take advantage of contextual information.

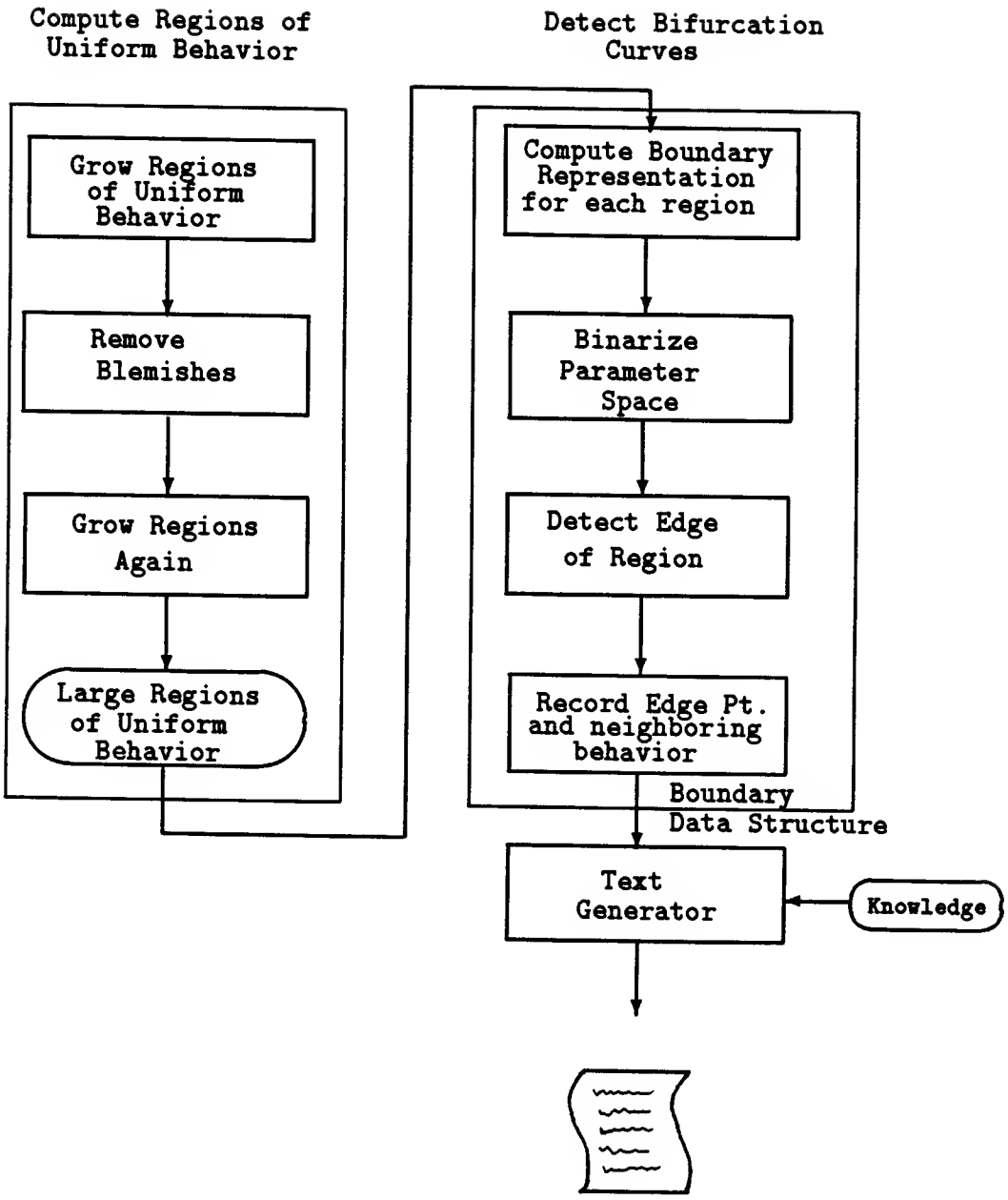

Figure 5.3: Strategy for Determining Boundary Types of a Region

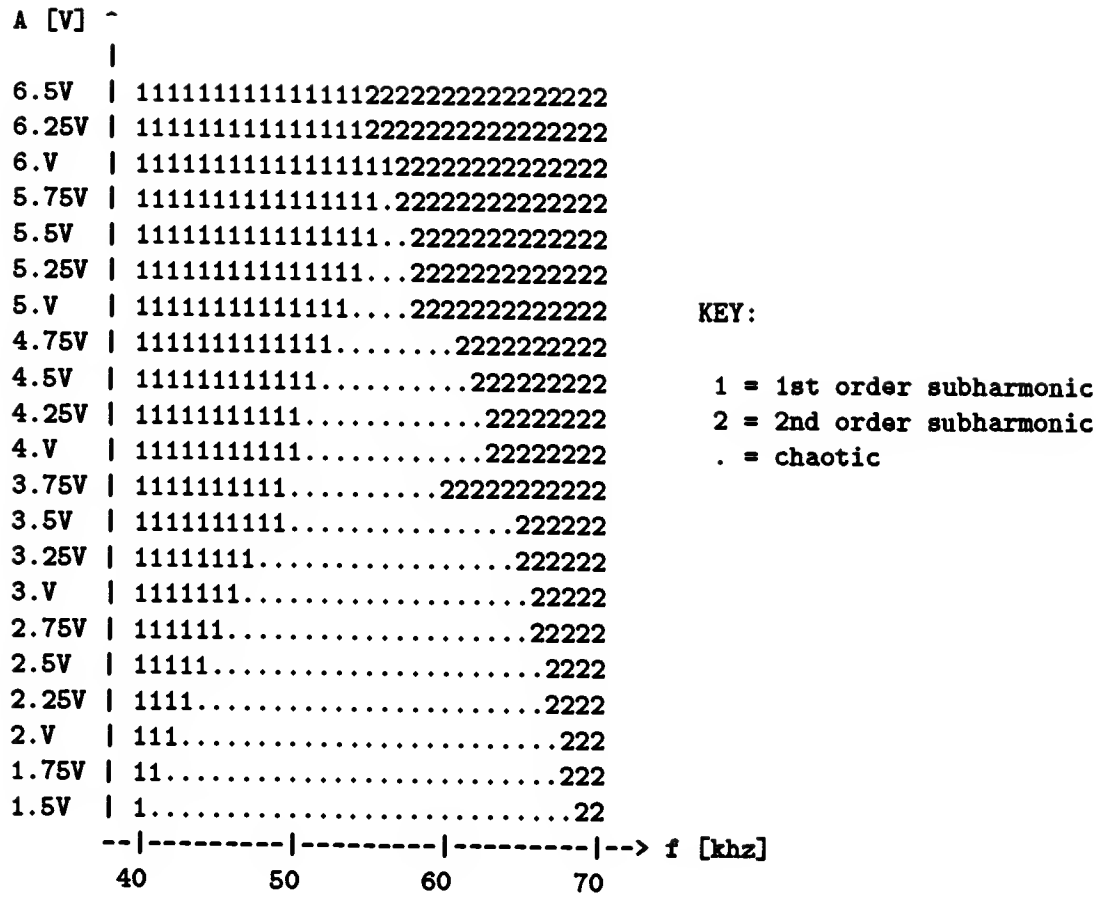

Figure 5.4: Cleaned up parameter space without blemishes.

subharmonic, and chaos.

Identifying large regions of very messy behavior

Now consider another parameter space diagram shown in figure 5.5 with a large region of first-order subharmonics centered about 5.25 volts and 46000 hertz, a large region of second-order subharmonics centered about 5.5 volts and 65000 hertz, a large region of chaos centered about 3 volts and 60000 hertz, and a large region of very messy behavior centered about 2.25 volts and 55000 hertz.

The high-level interpreter should be able to categorize this last, messy region as a

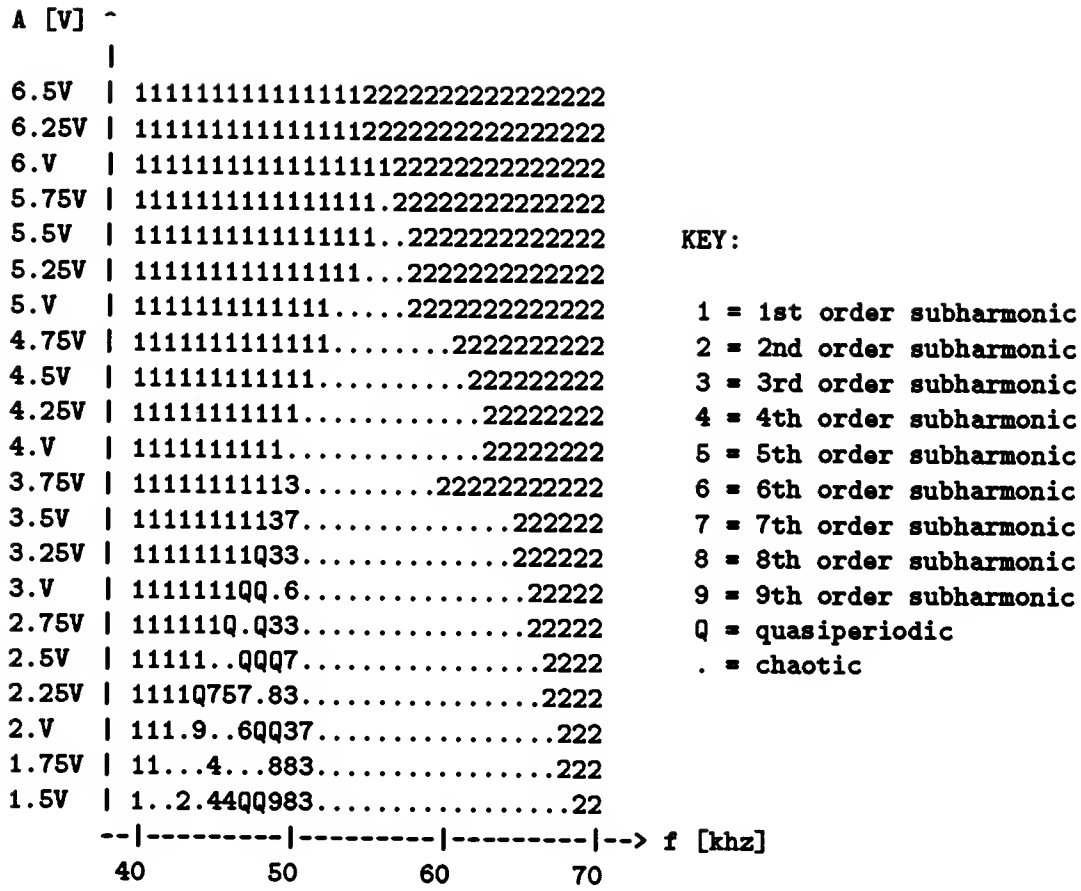

Figure 5.5: Parameter-space diagram with region of very messy behavior near 2.25 volts and 46000 hertz.

single region of very messy behavior, not as twenty-two tiny, distinct regions. Viewing these small regions as distinct would require the interpreter to compute the boundary types for each tiny region within this very messy area of the parameter space. Although this is possible, we would not gain very much in doing so. What the high-level interpreter does, then, is it converts all blemishes which are adjacent to another blemish into a new kind of region, "M", which stands for "messy" region. Thus, the parameter-space diagram shown in figure 5.5 would get transformed into the parameter space shown in Figure 5.6. After all the blemish points have been converted to "M," the Interpreter grows regions of uniform behavior once again. The result is the lumping together of points exhibiting qualitatively similar behavior. Notice how the messy region is now treated as a large, single region rather than many small regions.

The justification for categorizing these blemishes into a single region is as follows. An investigator examining the behavior of this nonlinear, electrical circuit would notice that there is a large region of first-order subharmonics at the lower frequencies, a large region of second-order subharmonics at the higher frequencies, a large region of chaos in the middle frequencies, and another region of very messy behavior at the lower frequencies and amplitudes. The details of this last region are not as crucial as the fact that the general behavior in the region is messy. In addition, from the practical point of view, identifying the different boundary types for each tiny blemish can be a tedious process. Instead, it is preferable to identify, in some rough, approximate sense, the large regions of uniform behavior and then to compute the boundary type for each.

5.2.2 Detecting bifurcation curves that border the regions of uniform behavior

At this point, the high-level interpreter has finished identifying the large regions of uniform behavior. The next task is to detect, for each region of uniform behavior, the bifurcation curves that border the region. In other words, we want to be able to say the region of chaos in figure 5.1 is bounded on one side by a C-1 bifurcation curve and on the other side by a C-2 bifurcation curve. To be able to make such a statement, the interpreter must first locate the edges of the regions in the parameter space.

Once again I have leveraged off of a computer vision technique called *binarization* to

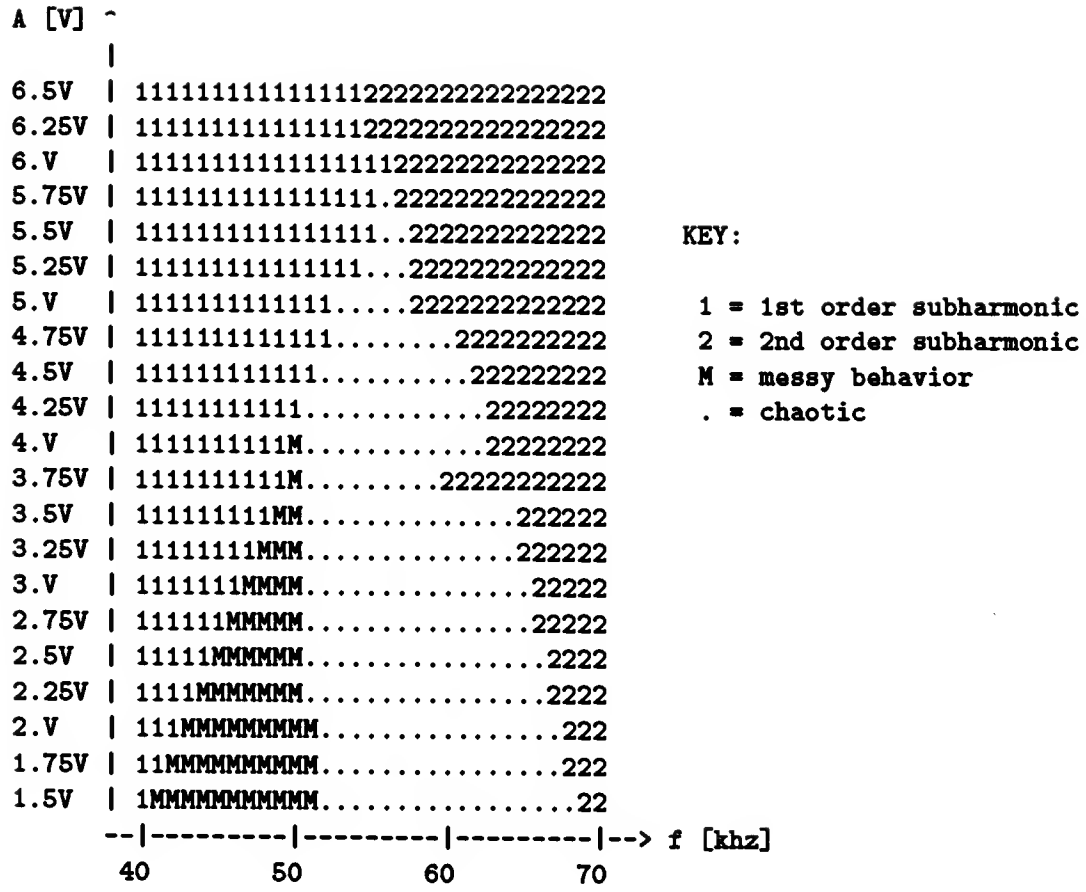

Figure 5.6: Parameter Space Diagram with Messy Region Indicated by "M."

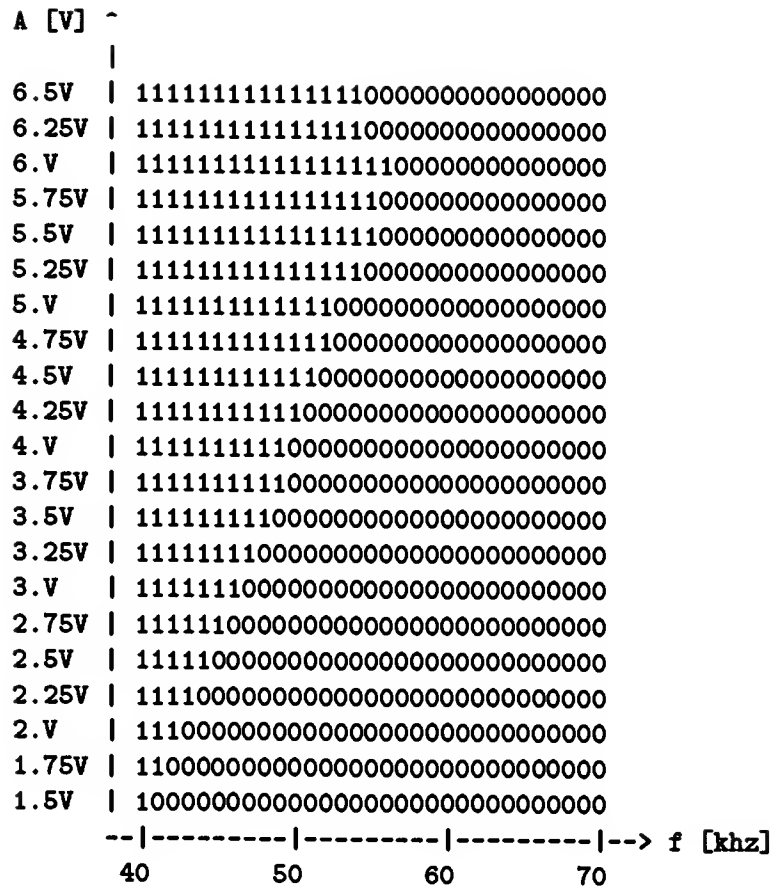

Figure 5.7: Binarized parameter-space diagram for the region of first-order subharmonics.

facilitate in edge detection[Hor86]. The strategy is as follows. Take a single region of uniform behavior and convert all points within this region to 1's and all other points to 0's. Next, scan the parameter space for $1 \rightarrow 0$ or $0 \rightarrow 1$ transitions. Any time you come across such a transition, record that point as an edge point. Also note the behavior of the adjoining point that caused this point to be an edge point. For example, referring again to figure 5.1, suppose we decide to binarize the region of first order subharmonics first. The result of binarizing the parameter space graph would produce a new graph as shown in figure 5.7.

We next scan the space in figure 5.7 searching for $1 \rightarrow 0$ transitions. Scanning the top row, we encounter our first $1 \rightarrow 0$ transition as the frequency is increased from 54000

hertz to 55000 hertz. Upon detecting this edge point, the interpreter records the point at 54000 hertz and 6.5 volts as an edge point and notices that the eastern point caused this point to be classified as an edge point. The program then looks up the behavior of this eastern point by referring to the original, unbinarized parameter space graph, notices it is a second-order subharmonic and records the behavior along with the edge point. Noting the behavior of the boundary-causing point is important because it allows the high-level interpreter to identify the boundary type.

In general, binarizing each region of uniform behavior has the advantage of simplifying the process of edge detection. The high-level interpreter need only search for $1 \rightarrow 0$ or $0 \rightarrow 1$ transitions. Once all the boundary points and the behaviors of the edge-causing neighbors have been recorded, we are left with a data structure containing a summary of all the boundary types of a region. Figure 5.8 illustrates the contents of a sample boundary data structure.

The particular example given in figure 5.8 came from the parameter-space graph shown in figure 5.1. Notice how the data structure accurately describes the behavior in the the parameter space. It says the region of first-order subharmonic, labelled "R1," is bounded by three types of regions: chaotic, messy, and second-order subharmonic. The region of messy behavior, labelled "R2," is bounded by two types of regions: chaotic, and first-order subharmonic. The region of chaotic behavior, labelled "R3," is bounded by three types of regions: messy, second-order subharmonic, and first-order subharmonic. And, the region of second-order subharmonic, labelled "R4," is bounded by two types of regions: chaotic and first-order subharmonic. Once the high-level interpreter has this data structure, generating the second paragraph of the textual explanation, which describes the bifurcation curves that border each region of uniform behavior, becomes straightforward.

5.3 Recognizing Typical Patterns in the Parameter Space

Finally, the purpose of the last paragraph of the textual explanation is to recognize and describe any interesting patterns observed in the parameter space, such as period-doubling cascades or period-adding cascades. As shown in figure 5.3, the two inputs

```
Specific Example:

(((SUBHARMONIC 1 "R1")
  ((MESSY "R2") (CHAOTIC "R3") (SUBHARMONIC 2 "R4")))
 ((MESSY "R2")
  ((CHAOTIC "R3") (SUBHARMONIC 1 "R1")))
 ((CHAOTIC "R3")
  ((MESSY "R2") (SUBHARMONIC 2 "R4") (SUBHARMONIC 1 "R1")))
 ((SUBHARMONIC 2 "R4")
  ((CHAOTIC "R3") (SUBHARMONIC 1 "R1"))))

General Format:

( <region-A>
   ( <bordering region A1> <bordering region A2>... <bordering region AN>)
  <region-B>
   ( <bordering region B1> <bordering region B2>... <bordering region BN>)
                                        .
                                        .
                                        .
  <region-Z>
   ( <bordering region Z1> <bordering region Z2>... <bordering region ZN>))
```

Figure 5.8: Boundary Representation Data Structure returned by High-Level Interpreter. Notice how the regions are numbered "R1" to "RN" to distinguish among multiple regions with the same kind of behavior.

to this module are: 1) the boundary representation data structure computed using the method outlined in the previous section, and 2) general knowledge about nonlinear dynamics. Given these two pieces of information, the high-level interpreter has enough information to identify and explain any interesting patterns. As an example, the program searches for a period doubling by tracing through the boundary data structure searching for the pattern $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow \ldots \rightarrow$ chaos. In reality, the interpreter does not actually trace through the full $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow \ldots \rightarrow$ chaos sequence. Instead, if it can successfully trace through four contiguous regions where the subharmonic order of the current region is twice the order of the previous region, then the interpreter declares the existence of a period-doubling cascade.

To expand the knowledge base of the interpreter, simply add new procedures to the high-level interpreter module. Each procedure must, of course, search out a given pattern such as $1 \rightarrow 2 \rightarrow 4 \rightarrow 8 \rightarrow 16 \rightarrow \ldots \rightarrow$ chaos for the period-doubling route to chaos, $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow \ldots \rightarrow$ chaos for the period-adding route to chaos, and periodic $\rightarrow$ chaotic $\rightarrow$ periodic $\rightarrow$ chaotic $\rightarrow$ periodic $\rightarrow$ chaotic $\rightarrow$ periodic ... for the alternating periodic-chaotic sequence. A particularly interesting route to chaos is called intermittency. To recognize an intermittent sequence to chaos, see if the system undergoes a transition from periodic behavior to chaotic behavior with occasional bursts of noise. Initially there should be long intervals of periodic behavior between short bursts of noise, but with increasing time the intervals between the bursts decrease; it becomes more and more difficult and finally impossible to recognize the regular oscillations of the periodic state[TS86].

5.4 Symbolic description

Finally, after producing the parameter-space diagram and the accompanying high-level text explanation, the program returns a symbolic summary of the circuit's behavior. Recall from chapter 1 that the symbolic descriptor may be passed on to another procedure for further analysis of the circuit's behavior. The other program might be interested in searching for more complicated patterns such as saddle-node bifurcations, flip bifurcations, or Hopf bifurcations[TS86]. Or the other program may be interested in locating

the regions of instability so as to advise the user to avoid operating the device in those regions. At any rate, figure 5.9 provides an abridged example of the symbolic descriptor generated by the program from the parameter-space graph shown in figure 5.1.

Specific Example:

```
( #(((CHAOTIC)                    ;; first region of uniform behavior
     (((VOLTAGE 1) (DRIVE-FREQUENCY 57000)) ;; pts. in chaotic region
     ((VOLTAGE 4) (DRIVE-FREQUENCY 56000))
                    .
                    .
     ((VOLTAGE 1) (DRIVE-FREQUENCY 50000))))
    ((SUBHARMONIC 2)              ;; second region of uniform behavior
     (((VOLTAGE 2) (DRIVE-FREQUENCY 68000)) ;; pts. in subharm 2 region
                    .
                    .
     ((VOLTAGE 2) (DRIVE-FREQUENCY 69000))
     ((VOLTAGE 1) (DRIVE-FREQUENCY 69000))))
    ((SUBHARMONIC 5) (((VOLTAGE 3) (DRIVE-FREQUENCY 56000))))
    ((SUBHARMONIC 7) (((VOLTAGE 3) (DRIVE-FREQUENCY 59000))))
                    .
                    .
    ((SUBHARMONIC 1)
     (((VOLTAGE 5) (DRIVE-FREQUENCY 51000))
      ((VOLTAGE 5) (DRIVE-FREQUENCY 50000))
      ((VOLTAGE 4) (DRIVE-FREQUENCY 50000))))
     ((SUBHARMONIC 9) (((VOLTAGE 4) (DRIVE-FREQUENCY 59000)))) )

   (((SUBHARMONIC 1 "R1")         ;; boundary representation
     ((MESSY "R2") (CHAOTIC "R3") (SUBHARMONIC 2 "R4")))
    ((MESSY "R2")
     ((CHAOTIC "R3") (SUBHARMONIC 1 "R1")))
    ((CHAOTIC "R3")
     ((MESSY "R2") (SUBHARMONIC 2 "R4") (SUBHARMONIC 1 "R1")))
    ((SUBHARMONIC 2 "R4")
     ((CHAOTIC "R3") (SUBHARMONIC 1 "R1")))) )
```

Figure 5.9: Sample symbolic description returned by the program. The first part of the symbolic descriptor describes the lumped regions of uniform behavior, while the second part provides information concerning the boundary representation.

Chapter 6

Conclusion

This thesis describes the design and implementation of a system that automatically measures and classifies the behavior of driven, nonlinear circuits and generates high-level, qualitative interpretations of their behaviors. In the concluding chapter, I would like to focus on the contributions and future work of this thesis.

6.1 Contributions of Thesis

6.1.1 Valuable Tool for Experimental Dynamicists and Other Investigators

At present the analyzer program is still at an early stage of development. It has only been tested on a few sample circuits —- the forced negative-resistance oscillator circuit[UA81], the series RLC circuit, and the devil's staircase circuit[KKC]. However, the results look promising. The analyzer program, along with the signal-processing module, the communication module, the measurement instruments, and the observation equipment provide a valuable tool for experimental dynamicists and other scientists studying the behavior of complex, nonlinear, electrical circuits. As mentioned in chapter 1, this system automates much of the time-consuming and tedious task of exploring the behavior of dynamical systems. Typically, in order to characterize the behavior of a nonlinear circuit, the investigator must select interesting parameter values with which to drive the circuit, measure the response, run a number of numerical computations (e.g. fast Fourier

transforms, autocorrelations and chirp z-transforms to help identify the behavior of the signal), and interpret the results. The results of these numeric computations are usually in the form of massive amounts of numerical data. Although the computer is very good at performing numeric computations, it is not very good at interpreting its qualitative content. Thus, it is largely up to the investigator to scan through the reams of data searching for relevant and important information. To have the analyzer program not only automatically set up and run the experiment, but produce a high-level qualitative description of the circuit's behavior, clearly facilitates the task of the investigator. As mentioned previously, these high-level, qualitative descriptions are typically what an investigator or experimental dynamicist is interested in. In fact, one of the aims of this project is to develop programs that can automatically generate descriptions of nonlinear dynamical systems similar to those found in published papers[AS87]. The content of the text from the high-level interpreter is based upon articles from the literature.

6.1.2 Integration of instrument environment with a high-level programming language and environment

This issue was touched upon briefly in chapter 2, but it is worth mentioning again here. The system provides a nice integration of the Scheme environment with the instrument environment. The advantage of such a combination is that it affords the investigator the flexibility of working in the Scheme environment, including such software packages as the signal-processing library, while also allowing access to a set of powerful measurement instruments.

Another advantage of integrating the Scheme environment with the instrument environment is that it hides from the user the low-level details of how to communicate with and operate each instrument. This may not seem like a very large advantage if you only have one or two instruments to use. However, if you need to use many instruments,[1] each of which has a whole different sequence of buttons to press to achieve the desired effect, you begin to appreciate a program which encapsulates this information. In our system, the communication module (refer to figure 2.3 of chapter 2) provides a nice interface be-

[1]The system for this thesis used six different instruments to explore the behavior of these nonlinear electrical circuits.

tween user and instruments. Thus, to perform a simple task like `record-a-signal`, the user need not open the owner's manual and start reading about the two-million-and-one options on the waveform recorder. Instead, the analyzer program automatically knows how to select the appropriate parameter values for optimal operation of the instrument, and the communication module can send the appropriate, cryptic HP-IB command to the instrument for the desired result. This saves the user a great deal of time and grief, especially as the number of instruments used in the system increases.

Finally, as mentioned in chapter 1, the integration of sophisticated measurement instruments with powerful numerical packages and software programs, which interpret the results of physical measurements and numeric computations, suggests new possibilities for test and measurement. No longer are scientists limited to low-level, control-the-voltmeter type commands. They now have access to more powerful, more *intelligent* tools, capable of generating such high-level statements as, "The forced negative-resistance oscillator undergoes a period doubling cascade as the driving frequency is increased from 26 khz to 78 khz while the amplitude is fixed at 5.25 volts."

6.2 Future Work

The Analyzer Program in combination with the rest of this system is a complex system. No doubt modifications and additions will be made to the system as it is tested on a greater number of nonlinear electrical circuits. However, I view this implementation as a successful *first* prototype. With regard to future work, I see several possibilities.

One possible improvement to the system would be to increase the sophistication of the pattern recognizer in the high-level interpreter so it can recognize more sequences than it currently does. At present, the pattern recognizer is still at a fairly early stage of development and can only identify a few patterns. However, as we include more knowledge about common sequences in nonlinear dynamics, we can increase the number and complexity of patterns the high-level interpreter can recognize. Chapter 17 of Thompson and Stewart[TS86] provides a catalog of additional patterns we may wish to include in our knowledge base such as intermittency, the U sequence, alternating periodic-chaotic sequence, etc.

In addition to including more patterns in the knowledge base, we may wish to take advantage of Feigenbaum's universal number when searching for period-doubling cascades. M. J. Feigenbaum discovered a *universal* solution common to all systems undergoing period doubling. For a given system, if we denote by Λ_n the value of the parameter at which its period doubles for the nth time, Feigenbaum's discovery can be summarized by the equation

$$\delta_n = \frac{\Lambda_{n+1} - \Lambda_n}{\Lambda_{n+2} - \Lambda_{n+1}}$$

where $\delta_n = \delta$ and $\delta = 4.6692016...$ for large n[TS86, Fei83]. Thus, if we know the parameter values at which the period doubled the previous two times, we can predict, approximately, the parameter value at which the period will double for the third time.

Another improvement to the system might be to include a feedback loop. The analyzer program, after performing a first pass analysis on a given region, should be able to automatically zoom in on regions of interest. For example, if the high-level interpreter sees a $1 \rightarrow 2 \rightarrow 4 \rightarrow 16 \rightarrow 32 \rightarrow \ldots \rightarrow$ chaos sequence, the program should zoom into the region between the 4th-order subharmonic and the 16th-order subharmonic to search for the missing 8th-order subharmonic. In so doing, the program behaves in a more intelligent manner because it has some idea of what it is looking for; and if the program doesn't find it, it looks more closely for the missing link.

In addition to augmenting the knowledge base and including a feedback loop, the use of the autocorrelation technique to help distinguish between quasiperiodicity and chaos needs to be explored. As mentioned in chapter 4, the use of this technique looks promising. However, the details of implementation have yet to be worked out.

For nonlinear systems, different initial conditions can lead to different steady state behaviors. Thus, to obtain truly reproducible parameter-space graphs, we need to initialize the state variables of the nonlinear circuit to the same state each time. The best way to implement this is by adding additional hardware to the circuit. In the case of Ueda's forced negative-resistance osciallator, for example, we might want add circuitry that automatically shorts out the capacitor and inductor prior to each test run.

A final suggestion about improvements to the system would be to have the resolver module use knowledge about commonly recognized patterns, such as the period doubling

cascade, to resolve conflicts between the conclusion of the time-signal analyzer and the power-spectrum analyzer. For example, if a period doubling cascade is unfolding in a given region, this piece of information may help the system extrapolate the behavior of a point within the region.

Bibliography

[33284] *3325A Synthesizer/Function Generator Operating and Service Manual.* Hewlett Packard Company, P.O. Box 69, Marysville, Washington 98270, 1984.

[51884] *5182A Waveform Recorder/Generator Operating and Programming Manual.* Hewlett Packard Company, 5301 Stevens Creek Boulevard, Santa Clara, CA 95050, 1984.

[AN83] Nasir Ahmed and T. Natarajan. *Discrete-Time Signals and Systems.* Reston Publishing, 1983.

[AS85a] Ralph H. Abraham and Christopher D. Shaw. *Dynamics – The Geometry of Behavior. Part 1: Periodic Behavior.* Aerial Press, 1985.

[AS85b] Ralph H. Abraham and Christopher D. Shaw. *Dynamics – The Geometry of Behavior. Part 2: Chaotic Behavior.* Aerial Press, 1985.

[AS85c] Ralph H. Abraham and Christopher D. Shaw. *Dynamics – The Geometry of Behavior. Part 3: Global Behavior.* Aerial Press, 1985.

[AS87] Harold Abelson and Gerald Jay Sussman. *The Dynamicist's Workbench: I. Automatic Preparation of Numerical Experiments.* Technical Report, Massachusetts Institute of Technology, May 1987. AI Memo 955.

[CNR81] J. Crutchfield, M. Nauenberg, and J. Rudnick. Scaling for external noise at the onset of chaos. *Physical Review Letters*, 46:933, 1981.

[CYY86] L. O. Chua, Y. Yao, and Q. Yang. Devil's staircase route to chaos in a nonlinear circuit. *International Joint Circuit Theory Applications*, 14:315–321, 1986.

[DBHL82] D. D'Humieres, M.R. Beasley, B.A. Huberman, and A. Libchaber. Chaotic states and routes to chaos in the forced pendulum. *Physical Review A*, 26:3483–3496, 1982.

[DLH88] David J. DeFatta, Joseph G. Lucas, and William S. Hodgkiss. *Digital Signal Processing: A System Design Approach.* John Wiley and Sons, 1988.

[Dov86] Webster P. Dove. *Knowledge-Based Pitch Detection.* PhD thesis, Massachusetts Institute of Technology, 1986.

[FCF*81] Harold Foehling, James P. Crutchfield, J. Doyne Farmer, Norman H. Packard, and Robert S. Shaw. On determining the dimension of chaotic flows. *Physica*, 3D:605–617, 1981.

[Fei83] Mitchell J. Feigenbaum. Universal behavior in nonlinear systems. *Physica*, 7D:16–39, 1983.

[Gle87] James Gleick. *Chaos: Making A New Science.* Viking, 1987.

[Har78] Fredric J. Harris. On the use of windows for harmonic analysis with the discrete fourier transform. *Proceedings of the IEEE*, 66:51–83, 1978.

[Hay64] Chichiro Hayashi. *Nonlinear Oscillations in Physical Systems.* McGraw-Hill, 1964.

[Hor86] Berthold Klaus Paul Horn. *Robot Vision.* The MIT Press, 1986.

[Hub83] B.A. Huberman. Mostly chaos. *Physica*, 118A:323–328, 1983.

[KC86] Michael Peter Kennedy and Leon O. Chua. Van der pol and chaos. *IEEE Transactions on Circuits and Systems*, CAS-33:974–980, 1986.

[KKC] Michael Peter Kennedy, Kenneth R. Krieg, and Leon O. Chua. The devil's staircase: the electrical engineer's: fractal.

[Lin81] Paul Linsay. Period doubling and chaotic behavior in a driven anharmonic oscillator. *Physical Review Letters*, 47:1349–1352, 1981.

[Lin88] Paul Linsay. Quasiperiodicity and chaos in a system with three competing frequencies. *Physical Review Letters*, 60:2719–2722, 1988.

[LLF82] A. Libchaber, C. Laroche, and S. Fauve. Period doubling cascade in mercury: a quantitative measurement. *J. Phys. Lett.*, 43:L211–L216, 1982.

[Lor63a] Edward N. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, 1963.

[Lor63b] Edward N. Lorenz. The mechanics of vacillation. *Journal of the Atmospheric Sciences*, 20:448–464, 1963.

[Lor64] Edward N. Lorenz. The problem of deducing the climate from the governing equations. *Tellus*, 16:1–11, 1964.

[MCK85] Takashi Matsumoto, Leon Chua, and Motomasa Komuro. The double scroll. *IEEE Transactions on Circuits and Systems*, CA-32:799–817, 1985.

[Moo87] Francis C. Moon. *Chaotic Vibrations: An Introduction for Applied Scientists and Engineers.* John Wiley and Sons, 1987.

[OS75] Alan V. Oppenheim and Ronald W. Schafer. *Digital Signal Processing.* Prentice Hall, 1975.

[PCFS80] Norman H. Packard, James P. Crutchfield, J. Doyne Farmer, and Robert S. Shaw. Geometry from a time series. *Physical Review Letters*, 47:712, 1980.

[RG75] Lawrence R. Rabiner and Bernard Gold. *Theory and Application of Digital Signal Processing.* Prentice Hall, 1975.

[Sha81] Robert Shaw. Strange attractors, chaotic behavior, and information theory. *Zeitschrift für Naturforschung*, 36a:80, 1981.

[Sha84] Robert Shaw. *The Dripping Faucet as a Model of Chaotic System.* Aerial, Santa Cruz, 1984.

[Spa82] Colin Sparrow. *The Lorenz Equations, Bifurcations, Chaos, and Strange Attractors.* Springer-Verlag, 1982.

[SR79] T.V. Sreenivas and P.F. S. Rao. Pitch extraction from corrupted harmonics of the power spectrum. *J. Acoust. Soc. Am.*, 65(1):223–228, 1979.

[SS75] Mischa Schwartz and Leonard Shaw. *Signal Processing: Discrete Spectral Analysis, Detection, and Estimation.* McGraw-Hill, 1975.

[Tak81] Floris Takens. Detecting strange attractors in turbulence. In D.A. Rand and L.S. Young, editors, *Lecture Notes in Mathematics*, page 336, Springer-Verlag, 1981.

[TPJ82] James Testa, José Pérez, and Carson Jefferies. Evidence for universal chaotic behavior of a driven nonlinear oscillator. *Physical Review Letters*, 48:517–522, 1982.

[TS86] J. M. T. Thompson and H. B. Stewart. *Nonlinear Dynamics and Chaos.* John Wiley and Sons, 1986.

[UA81] Toshisuke Ueda and Norio Akamatsu. Chaotically transitional phenomena in the forced negative-resistance oscillator. *IEEE Transactions on Circuits and Systems*, CAS-28:217–224, 1981.

[Wha71] A. D. Whalen. *Detectio n of Signals in Noise.* Academic Press, 1971.

[Wol83] Alan Wolf. Simplicity and universality in the transition to chaos. *Nature*, 305:182, 1983.

CS-TR Scanning Project
Document Control Form

Date : 7/27/95

Report # AI-TR-1125

Each of the following should be identified by a checkmark:
Originating Department:

☒ Artificial Intellegence Laboratory (AI)
☐ Laboratory for Computer Science (LCS)

Document Type:

☒ Technical Report (TR) ☐ Technical Memo (TM)
☐ Other:_____

Document Information

Number of pages: 88 (96 - IMAGES)
Not to include DOD forms, printer instructions, etc... original pages only.

Originals are:
☒ Single-sided or
☐ Double-sided

Intended to be printed as :
☐ Single-sided or
☒ Double-sided

Print type:
☐ Typewriter ☐ Offset Press ☒ Laser Print
☐ InkJet Printer ☐ Unknown ☐ Other: COPY ?

Check each if included with document:
☒ DOD Form (2) ☐ Funding Agent Form ☒ Cover Page
☒ Spine ☐ Printers Notes ☐ Photo negatives
☐ Other: _____

Page Data:

Blank Pages (by page number): _____

Photographs/Tonal Material (by page number): _____

Other (note description/page number):

Description : Page Number:
IMAGE MAP (1-88) PAGES #'ED 1-3, UN#'ED BLANK, 4-8, UN#'ED BLANK, 9-86
(89-93) SCANCONTROL, COVER, SPINE, DOD(2)
(94-96) TARGETS (3)

Scanning Agent Signoff:
Date Received: 7/27/95 Date Scanned: 7/28/95 Date Returned: 8/3/95

Scanning Agent Signature: Michael W. Cook

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AI-TR 1125
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER: AD-A 210919

4. TITLE (and Subtitle)

Summarizing Qualitative Behavior from Measurements of Nonlinear Circuits

5. TYPE OF REPORT & PERIOD COVERED: technical report

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)

Michelle Kwok Lee

8. CONTRACT OR GRANT NUMBER(s): N00014-86-K-0180

9. PERFORMING ORGANIZATION NAME AND ADDRESS

Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA 02139

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

Advanced Research Projects Agency
1400 Wilson Blvd.
Arlington, VA 22209

12. REPORT DATE: May 1989
13. NUMBER OF PAGES: 86

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

Office of Naval Research
Information Systems
Arlington, VA 22217

15. SECURITY CLASS. (of this report): UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

qualitative analysis
parameter-space graph
dynamical systems
symbolic description

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The process of exploring the behavior of nonlinear, dynamical systems can be a time-consuming and tedious process. In this thesis, I have written a program which automates much of the work of an experimental dynamicist. In particular, the program automatically characterizes the behavior of any driven, nonlinear, electrical circuit exhibiting interesting behavior below the 10 Mhz range. In order to accomplish this task, the program can autonomously select interesting input parameters, drive the circuit, measure its response, perform a set of numeric computations on the measured data, interpret

Block 20 cont.

the results and decompose the circuit's parameter space into regions of qualitatively distinct behavior. The output is a two-dimensional portrait summarizing the high-level, *qualitative* behavior of the nonlinear circuit for every point in the graph as well as an accompanying textual explanation describing any interesting patterns observed in the diagram. In addition to the graph and the text, the program generates a symbolic description of the circuit's behavior. This intermediate data structure can then be passed onto other programs for further analysis.

Scanning Agent Identification Target

Scanning of this document was supported in part by the **Corporation for National Research Initiatives,** using funds from the **Advanced Research Projects Agency** of the **United states Government** under Grant: **MDA972-92-J1029.**

The scanning agent for this project was the **Document Services** department of the **M.I.T Libraries.** Technical support for this project was also provided by the **M.I.T. Laboratory for Computer Sciences.**